AFIT/DS/ENG/98-02

Modified Multiple Model Adaptive Estimation (M^3AE) for Simultaneous Parameter and State Estimation

DISSERTATION

Mikel Mark Miller B.S., M.S.

Major, USAF

AFIT/DS/ENG/98-02

Disclaimer

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

AFIT/DS/ENG/98-02

Modified Multiple Model Adaptive Estimation (M^3AE) for Simultaneous Parameter and State Estimation

DISSERTATION

Presented to the Faculty of the Graduate School of Engineering of the Air Force Institute of Technology Air University In Partial Fulfillment for the Degree of

Doctor of Philosophy

Mikel Mark Miller B.S., M.S.

Major, USAF

Air Force Institute of Technology

Wright-Patterson AFB, Ohio

March, 1998

AFIT/DS/ENG/98-02

Modified Multiple Model Adaptive Estimation (M^3AE) For Simultaneous Parameter and State Estimation

Mikel Mark Miller B.S., M.S.

Major, USAF

Approved:

Peter S. Maybeck — 24 Mar 98
Dr. Peter S. Maybeck, Committee Chairman — Date

[signature] — 24 Mar 98
Dr. Stewart L. DeVilbiss, Committee member — Date

Mark E. Oxley — 24 Mar 98
Dr. Mark E. Oxley, Committee member — Date

[signature] — 24 Mar 98
Dr. Dennis C. Dietz, Dean's Representative — Date

Accepted:

[signature]
Dean Robert A. Calico, Jr
Dean, School of Engineering

Dedication

I dedicate this work to first and foremost to my Father in Heaven. Who, without His love and spiritual guidance, none of this would have happened. Next I dedicate this work to my fantastic wife, Colleen, whose love, prayers and support have been unceasing and have continued to increase during this effort. To my four wonderful children, Casey, Krista, Trevor, and Megan whose love and prayers helped me tremendously. Finally, I dedicate this work to my father and mother. Thanks mom and dad for doing a great job – This one's for you!!

Acknowledgments

I am thankful and indebted to so many people who have gone out of their way to assist me, in many different ways, to complete this effort. Most importantly, I would like to express my *deepest heartfelt gratitude* to Dr. Peter S. Maybeck, who has been a major influence in my life these past years. His spiritual and professional guidance have been greatly appreciated. He is a true *mentor*. His dedication to his students and the fine art of teaching is unmatchable. He kept me motivated through the tough as well as the good times. It was a true honor and privilege to work with him. Dr. Maybeck, I could not have done this without you – God Bless you! I would also like to extend my sincere thanks to the rest of my research committee, Dr. Stewart DeVilbiss and Dr. Mark Oxley, for helping me form this dissertation into a meaningful document – God Bless you all!

A special and sincere thanks go to Juan Vasquez, a fellow PhD student. His enthusiasm, focus and drive helped me get through more difficult times than I can count. We spent many weeks together to get the "basic" stuff working, and his insight and dedication were essential to my success. Juan – God Bless you!

Thanks to Mr. Stan Musick who provided key assistance in helping me to understand and use the MSOFE software – God Bless you!

My deepest thanks go out to my fellow Promise Keepers and St. Luke's Men's Fellowship, who have had me high on their prayer lists these past several years – God Bless you all!

Finally, I can't thank my wife and children enough for all their prayers, love and support. Especially Colleen, words cannot express my love and appreciation for her. She has always been there for me and has helped me to grow during this process. Colleen, Casey, Krista, Trevor, and Megan – Thank God for you and God Bless you.

Table Of Contents

List of Figures

Page

Page

List of Tables

Abstract

In many estimation problems, it is desired to estimate system states and parameters simultaneously. However, inherent to traditional estimation architectures of the past, the designer has had to make a trade-off decision between designs intended for accurate state estimation versus designs concerned with accurate parameter estimation. This research develops one solution to this trade-off decision by proposing a new architecture based on Kalman filtering (KF) and Multiple Model Adaptive Estimation (MMAE) techniques. This new architecture, the Modified-MMAE (M^3AE), exploits the benefits of an MMAE designed for accurate parameter estimation, and yet performs at least as well in state estimation as an MMAE designed for accurate state estimation. The M^3AE accomplishes the simultaneous estimation task by providing accurate state estimates from a single KF designed to accept accurate parameter estimates from the MMAE. Additionally, an M^3AE approximate covariance analysis capability is developed, giving the designer a valuable design tool for analyzing and predicting M^3AE performance before actually implementing the M^3AE and conducting a time-consuming full-scale Monte Carlo performance analysis. Finally, the M^3AE architecture is applied to two existing research examples to demonstrate the performance improvement over that of conventional MMAEs. The first example involves a simple second-order mechanical translational system, in which the system's natural frequency is the uncertain parameter. The second example involves a 13-state nonlinear integrated Global Positioning System/Inertial Navigation System (GPS/INS) system, in which the variance of the measurement noise affecting the GPS outputs, is the uncertain parameter.

Modified Multiple Model Adaptive Estimation (M^3AE) for Simultaneous Parameter and State Estimation

Chapter 1 - Introduction

1.1 Overview

This research focuses on adaptive filtering for the simultaneous estimation of both states and parameters for linear, dynamic, sampled-data systems. The proposed architecture employs two separate filters operating simultaneously to produce accurate state estimates under any foreseen system "operating condition." Under this new architecture, the parameter estimator is designed and tuned for estimating the system's uncertain parameters (for example, its current operating condition) and is optimized for distinguishing between possible hypothesized operating conditions (for example, typical applications involving parameter estimation are event detection and Failure Detection and Isolation). In contrast, the state estimator is designed and tuned for providing accurate state estimation, conditioned on the measurements and knowledge of the parameters provided by the parameter estimator. Figure 1 depicts this adaptive filtering concept (notice the forced separation between state and parameter estimation, versus treating the state-and-parameter estimation problem as one big nonlinear estimation problem). The measurements, $\mathbf{z}$, and parameter estimates, $\hat{\mathbf{a}}$, are used by the state estimator to generate $\hat{\mathbf{x}}$ under all feasible operating conditions.

Various techniques previously used for state and parameter estimation were researched as potential candidates for implementation in this new architecture. The four that are briefly described in Section 1.2 are the: Chi-Square test; Generalized Likelihood Ratio (GLR) testing; Multiple Model Adaptive Estimation (MMAE); and Distributed Kalman Filtering (DKF). Of those investigated, the

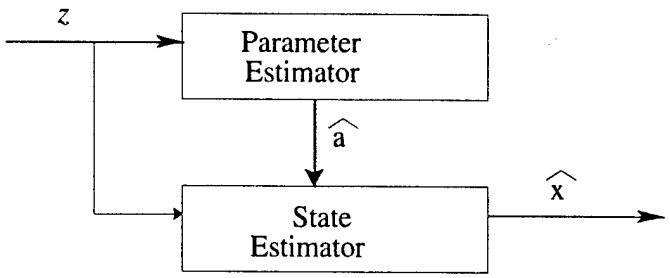


Figure 1. Simultaneous Parameter and State Estimation Using Separate Filters

MMAE technique was the strongest candidate and pursued in this research effort. A more detailed description of MMAE theory is presented in Chapter 2.

The Modified-Multiple Model Adaptive Estimation (M^3AE) technique has resulted from this research and is an enhancement in the evolution of MMAE architectures. MMAE architectures have been successfully used for both state and parameter estimation in the presence of uncertain parameters [2, 3, 14, 15, 21, 41, 46, 47, 54, 78]. The MMAE provides the designer a multiple-Kalman-Filter (KF) environment designed to provide accurate state estimation despite the uncertain parameters varying over a continuous region of the parameter space (though the MMAE conceptually assumes a discrete-valued or discretized parameter space, as explained in Chapter 2). MMAE's designed specifically for parameter estimation are typically concerned with applications involving event detection, such as detecting the onset of interference on a Global Positioning System (GPS) satellite signal [78], the tracking of targets despite the onset of maneuvers [32, 33, 41], or events associated with Failure Detection and Isolation (FDI) applications [2, 3, 5, 13–15, 21, 46, 53, 54, 56].

However, inherent to traditional MMAE architectures of the past, the designer has had to make a trade-off between an optimal design for state estimation versus a design concerned with accurate parameter estimation [35]. An M^3AE architecture exploits the benefits of an MMAE designed for

accurate parameter estimation, and yet performs at least as well in state estimation precision as an MMAE designed for accurate state estimation. The M^3AE architecture is straightforward and provides an excellent design option for designers concerned with applications in which both accurate parameter and state estimation is the design goal. Thus, the different classes of problems where the M^3AE is appropriate are those that require accurate estimation of both the parameters and the states, such as FDI and problems involving requirements for accurate state estimation during event changes affecting the system's operating condition. Figure 2 displays this architecture which offers enhanced design flexibility in optimizing each estimator for its intended purpose. The architecture involves a single KF designed to accept the parameter estimate provided by an MMAE designed for parameter estimation. Since the goal of the MMAE "block" is accurate parameter identification, the elemental filters are designed and tuned such that their resulting hypotheses are as distinguishable as possible from each other. This increases the MMAE's ability to detect event changes in the system accurately. Many MMAE options exist for this purpose. In addition to a "standard" fixed-bank MMAE, Lund's MMAE with Inter-Residual Distance Feedback (IRDF) technique [35], and Moving-Bank MMAE architectures [5, 19, 20, 23, 31, 46, 48, 75, 77] are strong candidates for detecting event changes in the system and are further discussed in Chapter 2.

Note that, the M^3AE architecture provides one solution to the simultaneous parameter and estimation challenge which preserves linearity of the state estimation portion of the problem by treating the parameters fundamentally differently from the states. Other methods include treating the unknown parameters as additional state variables, but this leads to a nonlinear estimation problem as solved approximately by extended Kalman filters or higher order nonlinear filters, but typically with nonnegligible biases in the parameter estimates [44, 45]. Furthermore, such a nonlinear filter formulation typically requires a priori parameter statistical information or direct physical knowledge about the unknown parameter. Thus, it is preferable *not* to treat parameters simply as additional

state variables to be estimated within the category of state-and-parameter estimators that do treat parameters differently from states, there are two different subcategories. First, there are algorithms with a single Kalman-like filter, from which a single residual vector is used as input to a parameter estimator, and then those estimated parameters are put back into the single state estimator. Maximum likelihood estimators and least squares estimators of this form have been used for many years [45]. However, it is difficult to provide quick and accurate responsiveness to true parameter changes with such an architecture. The second type of algorithm is based upon multiple models (i.e., multiple hypotheses of parameter values) and is composed of multiple Kalman filters in parallel, with residuals from *each* to be used to provide more rapid and precise tracking of changing real-world parameters. Therefore, the M^3AE is a strong candidate for simultaneous state and parameter estimation, since it inherently combines the best attributes of both forms of architectures within this preferred category of estimators.

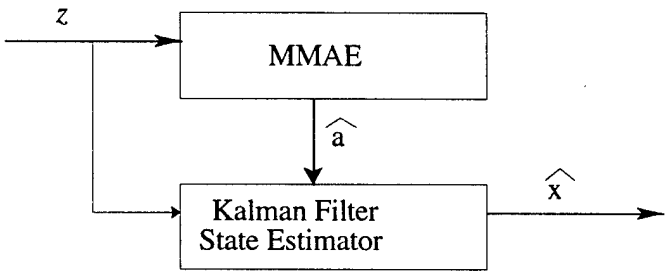


Figure 2. M^3AE – MMAE-Based Parameter Estimator and KF-Based State Estimator

Given the fixed bank MMAE structure chosen for this research, Lund's IRDF algorithm (whose primary purpose is to keep filter residuals of the individual elemental filters within the MMAE "distant" for the purpose of fast and reliable discrimination) is a strong candidate for implementation in this research, since the successful operation of the MMAE depends heavily on the distinguishability

of the models used in, and the tuning of, the elemental filters. This is appropriate for an MMAE producing a parameter estimate $\hat{\mathbf{a}}$, as proposed in the M^3AE architecture, but may not be as appropriate for an MMAE designed for state estimation (despite that being the intent of Lund's original work) since the state tracking capability of the MMAE may suffer [35]. Although this effort has focused on a fixed-bank MMAE architecture, further research using moving-bank MMAE architectures is highly recommended.

Finally, given an accurate parameter estimate describing the current system operating condition, the M^3AE's additional purpose is to estimate the system states accurately. Therefore, as shown in Figure 2, the measurements $\mathbf{z}$, and the MMAE-produced parameter estimate $\hat{\mathbf{a}}$, are simultaneously provided to a single KF which has been designed and tuned specifically for state estimation performance (note that the single state estimator could have different dimension and tuning characteristics than any of the elemental filters in the MMAE). Thus, the goal of simultaneous parameter and state estimation is achieved in a relatively straightforward manner.

In addition to the goal of providing accurate state estimation in the face of uncertain parameter variations, an additional objective of this research is to provide the designer with a useful tool for analyzing and predicting M^3AE performance. Typically, a covariance analysis tool has been available for analyzing and predicting performance for linear systems driven by white Gaussian noise, and whose available measurements are linear and corrupted by white Gaussian noise. One run of such a covariance analysis generates the time history of the covariance of the true estimation errors committed by a given Kalman filter in a specified "truth model" environment. Additionally, since the covariance expressions are independent of the actual measurement time history, a covariance analysis may be accomplished without actually running a simulation. Thus, one particular use for covariance performance (sensitivity) analysis is to perform Kalman filter tuning before applying it to real-time use [43]. Covariance analysis tools are essential in the initial design process since time

consuming multi-run Monte Carlo studies may be avoided until a viable design option is chosen and detailed performance analysis is required.

Designers employing MMAE-based systems have had filter-computed state estimation error and parameter estimation error covariance expressions available for making design decisions, but at least a single Monte Carlo run was required to generate the actual measurement histories and elemental filter probabilities required to solve those covariance expressions [44] (see Section 3.3 for further discussion). Any new and viable architecture should have a similar tool available to designers. As shown in Figure 3, a designer developing a system based on the M^{3}AE architecture may conduct an approximate covariance analysis of the M^{3}AE state estimates after a single Monte Carlo run on the MMAE-based parameter estimator within the M^{3}AE architecture. The single Monte Carlo run provides the time histories of the parameter estimates, $\hat{\mathbf{a}}(t_i)$, the MMAE-computed error covariance, $\mathbf{P_a}(t_i)$, and their associated elemental filter probabilities, $p_j(t_i)$. This information is then provided to the approximate M^{3}AE covariance analysis tool, which produces the required covariance analysis information to conduct a performance or sensitivity study of the M^{3}AE. The theory supporting this new approximate covariance analysis tool is presented in detail in Chapter 3.

As shown in Figure 3, the approximate covariance analysis tool allows efficient *iterative looping* to the left of the dashed line, before final full-scale Monte Carlo analysis is accomplished. The effectiveness of the approximate covariance analysis will be demonstrated by comparing it to the corresponding full-scale Monte Carlo analysis results from two applications presented in Chapter 4.

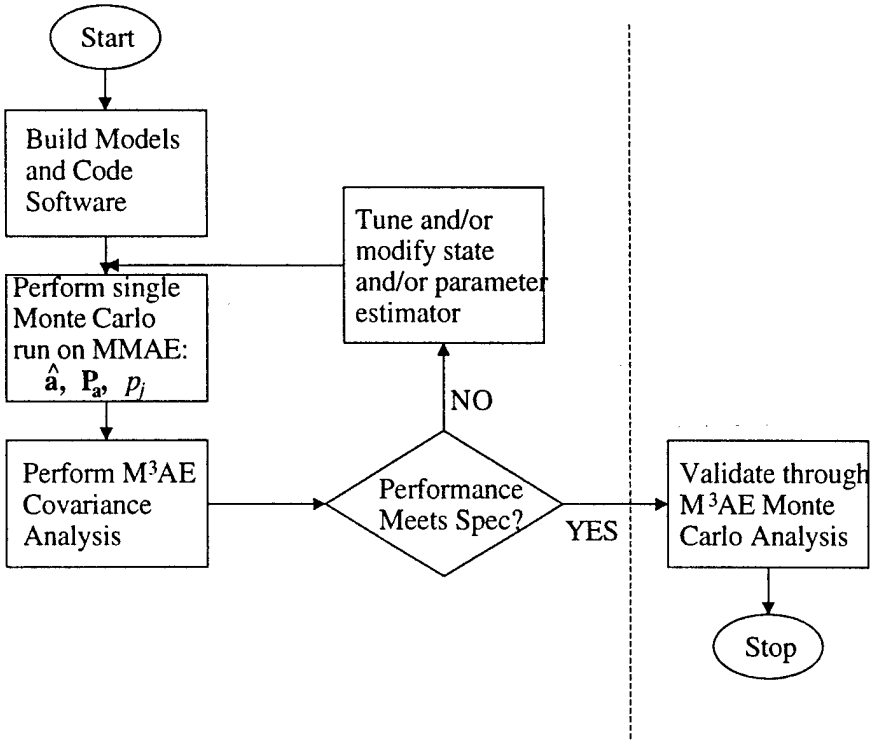


Figure 3. Performance Analysis Tool for M^3AE

1.2 Background

One common application area of parameter estimation and event detection is FDI, where the goal is to detect the onset of a failure and its associated failure type accurately. This section briefly discusses FDI and presents the four techniques mentioned earlier (Chi-Square, GLR, MMAE, and DKF) from an FDI perspective. This is done to provide a common frame of reference for comparing the techniques, even though the two examples presented in Chapter 4 are concerned with parameter estimation and event detection, not FDI explicitly.

Failure detection algorithms typically analyze multiple information sources to determine if a fault has entered a system. A fault could range from a bias on an incoming measurement to a hard failure of an entire subsystem. Each failure detection algorithm differs in function and complexity. They range from simple algorithms to detect the existence of failures in the system, to complex algorithms which detect, isolate and recover from faults. However, despite the differences in various algorithms, the fundamental objective of failure analysis is to determine failures based on discrepancies between different information sources [80].

Thus, failure analysis depends on the availability of multiple information sources providing redundant information, projected into the same reference frame. Multiple information is critical, since with only one source of information, failed operation is not easily distinguished from normal system operation. Additionally, if the information is not transformed into a common reference frame, there is no common basis for comparison [28]. Note, however, that the examples presented in Chapter 4 are not FDI-related and as such, do not rely on multiple information sources.

Identification of failure types is also critical to failure analysis and subsequent fault recovery. In general, a thorough failure mode analysis must be performed in any given application, in order to provide an adequate set of hypotheses to use as a basis for the FDI system. Analogously, with regard to this research, a thorough event analysis is required to serve as a basis for the elemental filter selections within the MMAE structure. Then, for effective fault recovery, corrective feedback, determined from failure mode analysis, is often required. For example, if a measurement bias is identified, corrective feedback enables the system to "remove" the bias for continued operation.

Given this basic understanding of failure analysis, this section presents the general theory behind four design options including the Chi-Square Test, GLR testing, DKF, and MMAE based techniques. Additionally, the drawbacks to using Chi-Square Test, GLR Testing, and DKF are presented,

compared to the benefits of an MMAE based architecture. Each technique is evaluated with respect to its ability to conduct effective FDI.

1.2.1 The Chi-Square Test

The following section parallels the discussion presented in [77]. The Chi-Square testing algorithm assumes a Kalman filter is used to blend the multiple information sources and calculates a random variable $\chi(t_k)$ based on the filter outputs [51, 64, 66, 68, 76]. The Chi-Square test uses information from Kalman filter measurement residuals, $\mathbf{r}(t_i) = \mathbf{z}(t_i) - \mathbf{H}(t_i)\hat{\mathbf{x}}(t_i^-)$ (where $\mathbf{H}(t_i)$ is the measurement matrix used within the filter and $\hat{\mathbf{x}}(t_i^-)$ is the filter's predicted state value at time t_i before the measurement at t_i is incorporated), to decide whether a failure has occurred. If the filter model matches the "truth" model, then the residuals will be a zero-mean white Gaussian process with known residual covariance, $\mathbf{A}(t_i) = \mathbf{H}(t_i)\mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i) + \mathbf{R}(t_i)$ (where $\mathbf{P}(t_i^-)$ is the conditional covariance of $\mathbf{x}(t_i)$ before the measurement $\mathbf{z}(t_i)$ is taken and processed, and where $\mathbf{R}(t_i)$ is the covariance of the measurement noise) [43]. If the residuals $\mathbf{r}(t_i)$ have larger magnitudes than anticipated by the filter-computed residual covariance $\mathbf{A}(t_i)$, a mismatch between the truth and filter-assumed model has occurred, implying the onset of a failure. These increased magnitudes may appear as a nonzero mean or a change in the covariance of the residuals. The chi-square random variable, $\chi(t_k)$, provides a test statistic that puts a quadratic penalty on variations in the residuals. Thus, given m-dimensional residuals, $\mathbf{r}(t_i)$:

$$\chi(t_k) = \sum_{i=k-N+1}^{k} \mathbf{r}^{\mathrm{T}}(t_i)\mathbf{A}^{-1}(t_i)\mathbf{r}(t_i) \tag{1}$$

is a Chi-Square random variable with Nm degrees of freedom, where N is the size of a window sliding across the residual values (used to make decisions based on the most recent N residuals). Note that this test relies only on information in the residuals.

If the system is operating normally (no failure – the H_0 hypothesis), then $\chi(t_k)$ should remain relatively small. However, if a failure occurs (the H_1 hypothesis), the statistics associated with $\mathbf{r}(t_i)$ should grow larger than anticipated through $\mathbf{A}(t_i)$, and $\chi(t_k)$ should increase in size. Thus, failure detection is accomplished by using a simple detection rule, based on a predetermined threshold value $T > 0$, of the form:

$$\begin{aligned} \chi(t_k) &> T \quad \Rightarrow \quad FAILURE \\ \chi(t_k) &\leq T \quad \Rightarrow \quad NO\ FAILURE \end{aligned} \tag{2}$$

Note that the size N of the averaging interval window and the threshold T are design parameters. It is the designer's goal to determine an acceptable trade-off between false alarms (declaring H_1 when actually H_0) and missed alarms (declaring H_0 when actually H_1), noting that as T increases, the probability of false alarm decreases, but the probability of missed alarms increases.

Previous work indicated the Chi-Square test as a highly effective and consistent failure detection technique [56, 76, 80, 81]. However, the algorithm is unable to perform failure isolation or recovery. Moreover, for this research effort "parameter" isolation and estimation is required for input into the state estimator.

1.2.2 Generalized Likelihood Ratio (GLR) Testing

The following section parallels the discussion presented in [77]. The GLR test is similar in nature to the Chi-Square test, but with the added benefit of failure detection and isolation. It is designed to distinguish between different types of failures and to estimate the magnitudes of the failure types [66, 74, 80, 81]. Like the Chi-Square test, the GLR test can exploit the residuals of

a Kalman filter as its basis for failure analysis and FDI. The GLR test also does a threshold test, however, it compares a generalized likelihood ratio function, $l(t_i, \theta)$ (generated from the ratio of the log-likelihood of possible hypotheses), to a predetermined threshold to determine whether or not a failure has occurred. It is derived from the GLR equations described below [56, 76, 80–82].

The hypotheses are established with a Kalman filter based on H_0 (no failure) and matched filters based on H_1 (a specific failure type added to the system). The ratio of the log-likelihood of the two hypotheses will be used to generate a generalized likelihood ratio function, $l(t_i, \theta)$, to be defined explicitly later in this section (see Equation (14)), which is used to declare failures via:

$$\begin{aligned} l(t_i, \theta) &> T \quad \Rightarrow \quad FAILURE \\ l(t_i, \theta) &\leq T \quad \Rightarrow \quad NO\ FAILURE \end{aligned} \tag{3}$$

If $l(t_i, \theta)$ is less than the predetermined threshold, T, then H_0 is declared true. Similarly, H_1 is declared true if $l(t_i, \theta)$ is greater than T. The parameter θ is the unknown time of the failure. The remainder of this section describes the derivation of the GLR algorithm for a single, step failure.

The following are the model equations upon which a Kalman filter might be based:

$$\mathbf{x}(t_i) = \mathbf{\Phi}(t_i, t_{i-1})\mathbf{x}(t_{i-1}) + \mathbf{G}_\mathrm{d}(t_{i-1})\mathbf{w}_\mathrm{d}(t_{i-1}) \tag{4}$$

with discrete measurements described by:

$$\mathbf{z}(t_i) = \mathbf{H}(t_i)\mathbf{x}(t_i) + \mathbf{v}(t_i) \tag{5}$$

The matching filters are designed for failure detection, not state estimation, and are based upon:

$$\mathbf{x}(t_i) = \mathbf{\Phi}(t_i, t_{i-1})\mathbf{x}(t_{i-1}) + \mathbf{G}_\mathrm{d}(t_{i-1})\mathbf{w}_\mathrm{d}(t_{i-1}) \tag{6}$$

$$\mathbf{z}(t_i) = \mathbf{H}(t_i)\mathbf{x}(t_i) + \mathbf{v}(t_i) + \mathbf{d}(t_i)n(t_i, \theta)\nu \tag{7}$$

where

$\mathbf{d}(t_i)$	=	failure vector
$n(t_i, \theta)$	=	failure function
ν	=	unknown size of the failure
θ	=	unknown time of the failure

Comparison of Equations (5) and (7) indicate that the matching filter characterizes failures by modeling them as variations in the actual measurements beyond the variations caused by the dynamics of the system or measurement noise, as indicated by the failure offset term, $\mathbf{d}(t_i)n(t_i,\theta)\nu$. Although the failure is modeled as a bias on the measurement, this model can also represent changes in the states caused by real world anomalies. The failure function term, $n(t_i,\theta)$, indicates the time of the failure onset, θ, within a predetermined sliding "window" of time, and the type of failure that has occurred; i.e., ramp offset, step offset, etc. A sliding window of predetermined length is used to avoid a growing set of hypotheses and slides in time to cover all the data collected (as the window slides in time, the oldest data in the window is discarded as the new data enters the filter – maintaining the same total amount of information). The ν term is the magnitude of the failure and can be estimated by the GLR algorithm and can also be used for corrective feedback. The column vector, $\mathbf{d}(t_i)$, specifies which of the measurement signals has the failure. In general, the likelihood ratio function, $l(t_i,\theta)$, is based on maximum likelihood estimates of θ and ν. The goal of the GLR algorithm is to identify the failure signal by recognizing variations in the residuals from their normal operating values.

The Kalman filter residuals $\mathbf{r}(t_i)$ are defined as

$$\mathbf{r}(t_i) = \mathbf{z}(t_i) - \mathbf{H}(t_i)\hat{\mathbf{x}}(t_i^-) \tag{8}$$

and the residuals for each hypothesis are described by

$$\begin{aligned} \mathrm{H}_0 &: \mathbf{r}(t_i) = \mathbf{r}^0(t_i) \\ \mathrm{H}_1 &: \mathbf{r}(t_i) = \mathbf{r}^0(t_i) + \mathbf{g}(t_i,\theta)\nu \end{aligned} \tag{9}$$

When the system is operating under normal conditions, the Kalman filter tracks the true states, and $\mathbf{r}^0(t_i)$ is zero-mean white Gaussian noise with covariance $\mathbf{A}(t_i) = \mathbf{H}(t_i)\mathbf{P}(t_i^-)\mathbf{H}(t_i)^{\mathrm{T}} + \mathbf{R}(t_i)$. When a failure occurs, a signal of unknown magnitude, $\mathbf{g}(t_i,\theta)\nu$, will be present in the residuals.

This signal is the failure residual offset and found through

$$\mathbf{g}(t_i,\theta) = \mathbf{H}(t_i)\mathbf{f}(t_i,\theta) + \mathbf{d}(t_i)n(t_i,\theta) \tag{10}$$

where the recursive failure quantity $\mathbf{f}(t_i,\theta)$ is given by

$$\mathbf{f}(t_{i+1},\theta) = \mathbf{\Phi}(t_{i+1},t_i)\left[\mathbf{I} - \mathbf{K}(t_i)\,\mathbf{H}(t_i)\right]\mathbf{f}(t_i,\theta) - \mathbf{\Phi}(t_{i+1},t_i)\mathbf{K}(t_i)\mathbf{d}(t_i)n(t_i,\theta) \tag{11}$$

Note that the GLR algorithm is a function of the overall system behavior ($\mathbf{\Phi}$ and $\mathbf{H}$) and Kalman filter gain $\mathbf{K}$ as shown in Equations (10) and (11). If the failure is assumed to occur at the beginning of the sliding window discussed earlier, then Equations (10) and (11) can be simplified by setting $n(t_i,\theta) = 1$ for all t_i:

$$\mathbf{g}(t_i) = \mathbf{H}(t_i)\mathbf{f}(t_i) + \mathbf{d}(t_i) \tag{12}$$

$$\mathbf{f}(t_{i+1}) = \mathbf{\Phi}(t_{i+1},t_i)[\mathbf{I} - \mathbf{K}(t_i)\mathbf{H}(t_i)]\mathbf{f}(t_i) - \mathbf{\Phi}(t_{i+1},t_i)\mathbf{K}(t_i)\mathbf{d}(t_i) \tag{13}$$

The primary reason for this simplification is to reduce the computational burden associated with calculating several GLRs based on different values of θ. However, the consequence of this simplification is a delay in detecting the failure caused by waiting for the failure to reach the beginning of the sliding window. The Kalman filter outputs combined with the matching filter model determines the magnitude of the generalized likelihood ratio function defined as

$$l(t_i,\theta) = \frac{S^2(t_i,\theta)}{C(t_i,\theta)} \tag{14}$$

where

$$S(t_i,\theta) \;=\; \sum_{j=1}^{i} \mathbf{g}^{\mathrm{T}}(t_j,\theta)\mathbf{A}^{-1}(t_j)\mathbf{r}(t_j) \tag{15}$$

is essentially the correlation of the observed residuals with the abrupt change signatures given by $\mathbf{g}(t_i,\theta)$ for the different hypothesized types and times of occurrence. Furthermore,

$$C(t_i,\theta) \;=\; \sum_{j=1}^{i} \mathbf{g}^{\mathrm{T}}(t_j,\theta)\mathbf{A}^{-1}(t_j)\mathbf{g}(t_j,\theta) \tag{16}$$

is interpreted as the amount of information present in $\mathbf{z}(t_1), ..., \mathbf{z}(t_i)$ when an abrupt change occurs at time θ. In both Equations (15) and (16), $\mathbf{A}(t_j)$ is given by

$$\mathbf{A}(t_j) = \mathbf{H}(t_j)\mathbf{P}(t_j^-)\mathbf{H}^{\mathrm{T}}(t_j) + \mathbf{R}(t_j)$$

and the maximum likelihood estimate (MLE) of the unknown magnitude of the failure, ν, is found by

$$\hat{\nu}(t_i, \theta) = \frac{S(t_i, \theta)}{C(t_i, \theta)} \tag{17}$$

The residual covariance $\mathbf{A}(t_j)$ and the residuals are combined in Equations (10) and (11) or Equations (12) and (13) to form a linear combination of the residuals $S(t_i, \theta)$ and a deterministic value $C(t_i, \theta)$ defined in Equations (15) and (16). Finally, the decision rule given by Equation (3) is used to determine the system's failure condition.

Thus, the GLR algorithm involves a single Kalman filter, a matched filter, and the likelihood calculation. It determines failures by observing changes in the filter residuals and calculates the likelihood of each possible event by correlating the residuals with the corresponding failure signature. Figure 4 depicts an example of a multiple GLR test with a bank of matched filters designed for the no failure and step failure modes. The Kalman filter provides its residuals to the bank, wherein each filter is tuned to a certain type of failure mode. Each matching filter's output is tested against a hypothesis, H_k, corresponding to each hypothesized failure mode. An MLE is computed for each specific hypothesis. Each MLE is fed into the common test logic algorithm, similar to Equation (3), to determine the correct hypothesis.

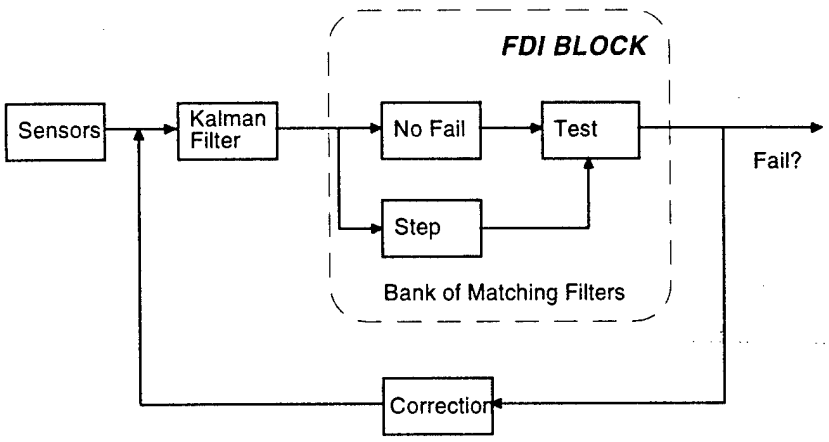


Figure 4. Multiple GLR Testing

One of the key benefits of the GLR test is the need for only one Kalman Filter. Additionally, only one matching filter is required for each failure type, since the algorithm estimates unknown variables, such as the magnitude of the failure type, in the FDI process. This is a great computational load benefit, especially in comparison to other multiple model techniques.

Even though GLR has been successfully applied to a wide variety of FDI applications [56, 66, 76, 80, 81], there are some limitations which include:

1. It has difficulty handling any parametric changes which may occur in the model due to changes in the system's operating condition. Thus, it is difficult for the GLR to model the dynamic nature of a system and its sensors to represent their behavior in the presence of failures. Therefore, typical GLR tests lack robustness since they are unable to detect parametric changes while looking for additive changes.

2. While detection of abrupt changes is a GLR's strength, detection of ramp failures is difficult.

An unacceptable time delay may occur since the ramp-corrupted signal is slowly moving away from the desired signal and takes more time to cross the failure threshold [76]; thus, depending on the length of the sliding window, the GLR test may fail to detect the failure. An additional drawback is that 'windowing' the estimate of θ leads to a direct reduction in the accuracy of the estimate of the size of the failure, ν.

3. Windowing may also cause an unacceptable delay in the identification of failures.

A direct comparison between the M^3AE and GLR/chi-square was not accomplished in this research effort. However, White [78] accomplished an indirect comparison of conventional MMAE-based techniques with a GLR/chi-square based technique applied to an FDI study performed by Vasquez [76] on a GPS/INS based system. The following observations were made:

1. Both methods were effective at detecting interference failures (represented by increased measurement noise variances), but the GLR/chi-square based scheme suffered from much larger time delays as compared to the MMAE.

2. The GLR/chi-square scheme also experienced large time delays when returning to a nominal no-fail declaration after receiving a large amount of interference, as compared to the MMAE.

3. Additionally, the GLR/chi-square algorithm suffered from its inability to detect/identify ramp failures adequately.

In summary, the GLR/chi-square failure testing scheme experienced unacceptable time delays compared to the MMAE-based techniques, especially in the face of bias-like failures.

1.2.3 Multiple Model Adaptive Estimation (MMAE)

The Kalman filter's ability to produce accurate estimates of the true states of a physical system is limited by how well its internal dynamics model adequately describes the true system dynamics.

This implies that the correct values of the parameters which describe the dynamic system (for example, the coefficients of the system differential equations) must be embedded in the filter model; however, some of these parameters are often unknown to the designer and/or changing in time. One method of dealing with this situation is through Multiple Model Adaptive Estimation (MMAE) [2,3, 15,22,36,44,46]. A promising alternative for FDI rests in the application of the MMAE technique. MMAE, like GLR and Chi-Square tests, exploits Kalman filter residual information, but it does so using multiple Kalman filters, each based on a specific hypothesis. The major strength of MMAE rests in its rapid response to changes in the real world. The MMAE algorithm [2,3,14,15,19,20,22, 22,30,33,35,36,38,41,44,45,49–51,53,54,57,68–70] naturally exploits the capabilities of parallel processing. By running multiple filters in parallel, residual information at each update is used to reconfigure the system rapidly to failures. Unlike the GLR algorithms which implement only a single Kalman filter (and model failures using matching filters), the MMAE employs multiple Kalman filters to model the dynamic nature of the system (and its sensors) and to represent performance in the presence of specific hypothesized parameter (failure) conditions. However, an inherent tradeoff exists between accurate parameter estimation, and accurate state estimation. Specifically, if one tunes the MMAE bank for optimal parameter estimation, then these same tuning values will often result in less than optimal state estimation. This is addressed in Chapter 2. Many researchers have focused on exploiting the capability of MMAE to provide state and parameter estimation as well as attempting to blend them ideally [2,3,5,14,15,17–22,26,27,32–35,41,43–47,50,53,54,57,67–73].

The MMAE's basic structure is shown in Figure 5. It is composed of a bank of Kalman filters, called elemental filters, which each use their own unique model to develop an estimate of the current states, $\hat{\mathbf{x}}_j$, independent of the other filters. Each elemental filter then uses this estimate, along with the current measurement, $\mathbf{z}$, to form its residual, $\mathbf{r}_j$ — the difference between the measurement and the filter's prediction of the measurement before it arrives, based on its hypothesized model. The

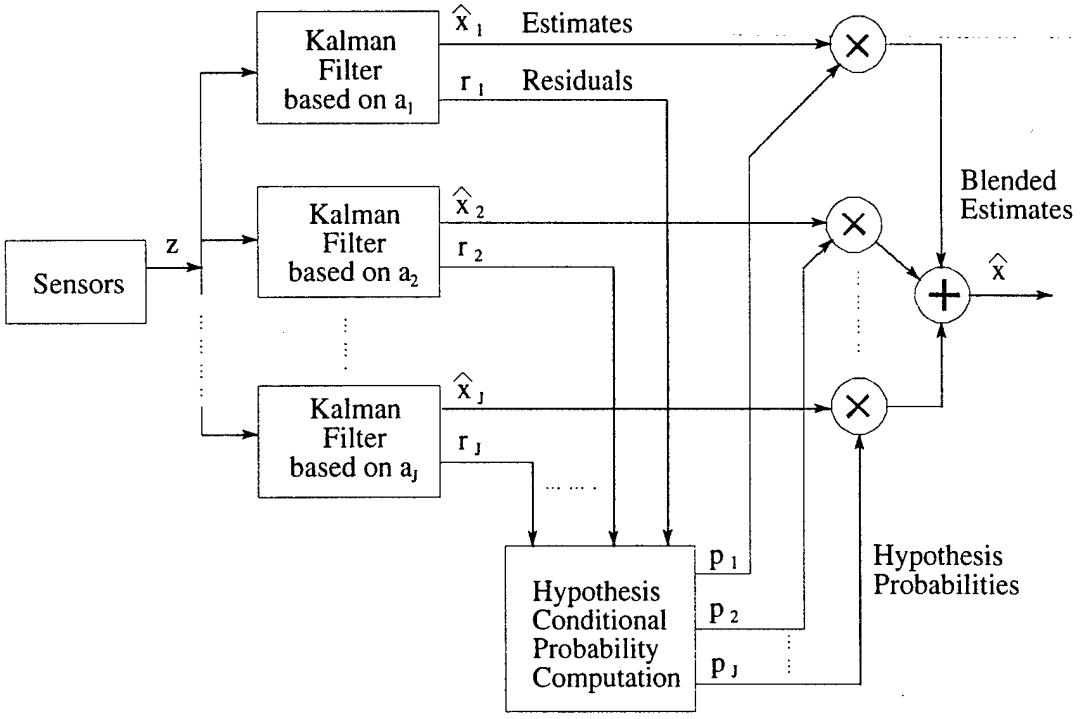


Figure 5. Multiple Model Adaptive Estimation Algorithm

residuals from the filters are used by the hypothesis testing algorithm as a relative indication of how close each elemental filter model is to the true model; the smaller the residual (or scaled residual, as discussed next), the closer that filter model represents the true model. The hypothesis testing algorithm first scales the residuals to account for anticipated magnitudes under its presumed hypothesis being correct by forming $\mathbf{r}_j^{\mathrm{T}}(t_i)\mathbf{A}_j^{-1}(t_i)\mathbf{r}_j(t_i)$ (where $\mathbf{A}_j(t_i) = \mathbf{H}_j(t_i)\mathbf{P}_j(t_i^-)\mathbf{H}_j^{\mathrm{T}}(t_i) + \mathbf{R}_j(t_i)$ is the residual covariance as computed by the j-th elemental filter), and then computes the conditional probability, p_j, for each of the hypotheses modeled in the bank of Kalman filters, conditioned on the measurement history observed up to that time (see equations in the Chapter 2). Each filter's state estimate, $\hat{\mathbf{x}}_j$, is blended together through a probability weighted average based on the conditional probability, p_j, of that filter modeling the true system operating condition. This blending allows for partial failures in a sensor or combinations of failure types, or parameter values between the discrete parameter point values used to define each hypothesis. A high probability of almost one indicates that a filter is extremely accurate in its modeling and will almost completely determine the final blended estimate, while all other model estimates will receive almost zero weighting.

From a parameter estimate standpoint, monitoring the elemental filter residuals or the conditional probabilities will provide insight into the operating condition of the system. Additionally, it will be shown in Section 2.1 that the parameter estimate $\hat{\mathbf{a}}$ can be found directly via a probability-weighted average of the discrete $\mathbf{a}_j$ values associated with each elemental filter, as a best estimate of the current system operating condition. It is important to note, that the number of elemental filters will affect the granularity of the parameter estimation and ultimately the accuracy of the state estimation.

A benefit to the MMAE structure, as opposed to a GLR, is its inherent ability to handle ramp-type failures, even though its elemental filters might only be based on *constant*-offset hypotheses.

This is true since, as the ramp magnitude grows in time, blended estimates should indicate the effects of the actual ramp on the various elemental filters.

Additionally, a significant benefit of MMAE is that the bank of Kalman filters, if well designed, can span the failure space. As long as the system is affected by a failure within the scope of the failure field, the filter is able to detect the occurrence of that failure; therefore, within the scope of the failure space, MMAE is a very effective FDI algorithm [44, 47, 49, 53, 54].

A major drawback of the MMAE is the potential for a large number of Kalman filters to span a realistic failure space adequately, creating a computational burden. If the MMAE is not implemented with a spanning set of filters, pertinent failure modes may not be modeled, resulting in poor residuals and limited accuracy of the blended state estimates. The same effect is possible if the discretization of the failure space is too coarse.

The MMAE concept and the various methods used to enhance MMAE performance will be explained in greater depth in Chapter 2.

1.2.4 Distributed Kalman Filtering (DKF)

An alternative to the stand-alone or "centralized" Kalman filter is the distributed Kalman filter (DKF). This section focuses on the DKF method developed by Carlson [8–10, 37]. The DKF, also called the federated filter, is another parallel structure like MMAE and a brief comparison of these algorithms will be presented after the basic structure of the DKF is introduced. DKF is based on simple and effective "information-sharing" methodology. The basic procedure involved in this information-sharing is [37]:

1. Divide the total system information among several component filters.
2. Perform local time propagation and measurement update processing (adding local sensor information when available).

3. Recombine the updated local information into a new total sum.

For example, in the multi-sensor navigation system shown in Figure 6, sensor measurements are sent to local filters (LF), which operate autonomously and possibly at different data rates. Each local filter receives sensor data from a "main sensor" such as the INS and an independent sensor such as the radar altimeter. Thus, this local filter holds part of the total system information. The remainder of the system information is contained in a master filter (MF). The information from the local filters is then "fused" with information in a master filter to form a full solution based on all of the system information. Depending on the DKF configuration being used, the MF *may* provide feedback to the LF's by resetting the initial conditions of the LF's. This reset may or may not contain the fused solution. The goal is to duplicate the performance of a single centralized filter for the same problem.

The advantages of the federated filter's information sharing technique are [37]:

- The distributed nature of the federated filter allows parallel processing in the local filters, possibly increasing total system throughput (measurements processed per second).
- System throughput is also increased by using local filters for data compression.
- System reliability is improved by maintaining multiple local solutions usable as backups. If a local filter gets corrupted, information from other local filters or the master filter can be used to restart the corrupted local filter.
- The federated filter allows the ability to detect difficult sensor faults, in particular slowly-deteriorating sensor data or 'soft faults', which may be difficult to do in the centralized Kalman filter. Moreover, in the event of undetected faults, estimates in other local filters do not get corrupted.
- System development, test and maintenance costs can be reduced by using multiple local filters, since the local filters should be of lower dimension and complexity than one inclusive centralized Kalman filter. Furthermore, adding a new sensor simply involves developing a new local filter and integrating it with the master filter, versus upgrading one, typically complex, centralized Kalman filter.

In Figure 6, there are s local filters and one master filter, totaling $r = s + 1$ filters. Note that each sensor sends measurements to a local Kalman filter which can operate independently as shown, or with information 'feedback' from the MF to each LF (not shown for simplicity).

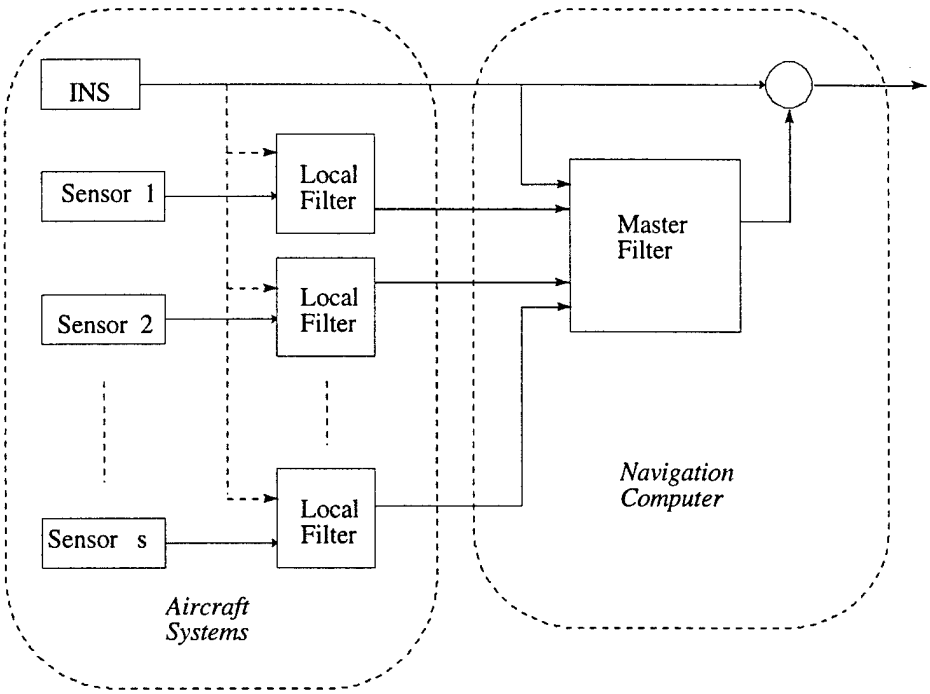


Figure 6. Federated Filter (DKF) General Structure For Aided Inertial Navigation Systems

When comparing MMAE and DKF it is important to note the following differences

- The elemental Kalman filters in an MMAE process all the measurements in the system and differ from one another according to their discrete assumed values of the parameter vector $\mathbf{a}$. The local Kalman filters in a DKF process the measurements from a "main sensor" and an independent sensor and differ from each other according to their components of the measurement vector $\mathbf{z}$.
- The structure of the DKF master filter differs from a standard Kalman filter as it only propagates its state estimates and covariances using inputs from the local filters – as opposed to making updates based on the raw sensor data. In fact, depending on which DKF information fusion scheme is employed (see [9, 10] for details of the various fusion schemes), the MF may not propagate anything and may simply be used to fuse the LF solutions.

The remainder of this section provides a brief description of Carlson's development which applies information-sharing principles in its use of multiple LF's and one MF [8]. Let the full (global) filter solution (as would be computed by a single centralized Kalman filter) be described by the state vector $\hat{\mathbf{x}}_f$ and the covariance matrix $\mathbf{P}_f$. Analogously, let the $k^{th}(k = 1, 2, ..., s)$ local filter solution be denoted by $\hat{\mathbf{x}}_k$ and $\mathbf{P}_k$, and the master filter solution by $\hat{\mathbf{x}}_m$ and $\mathbf{P}_m$. Then two key assumptions are made concerning the following DKF development: 1) the measurement errors from each of the sensors are statistically independent; and 2) the errors in LF and MF solutions are statistically independent.

These assumptions seem rather bold and quite probably impossible to verify in a real-world application, especially since the MF is fed by all the LF's and it will thus have errors that will generally be correlated with the LF errors. Additionally, when using error state space Kalman filters as in the GPS/INS example in Chapter 4, the above assumptions imply that the errors in the difference measurements are independent. This independence is difficult to accept since the filters are utilizing the same measurement sources for the difference measurements, resulting in correlation between the errors. However, in practice, DKF algorithms have yielded useful performance despite the violation of these assumptions [8, 10].

Now, if it is also assumed that the LF's and MF all have the same state dimensionality (this is not a practical assumption as the filters may be chosen to have a different number of states, but is

used here to demonstrate the general idea) and given that assumptions 1 and 2 above hold, the LF and MF solutions can be optimally combined by the following additive information algorithm [9, 10]:

$$\begin{aligned} \mathbf{P}_f^{-1} &= \mathbf{P}_m^{-1} + \mathbf{P}_1^{-1} + \mathbf{P}_2^{-1} + \ldots + \mathbf{P}_s^{-1} \\ \mathbf{P}_f^{-1}\hat{\mathbf{x}}_f &= \mathbf{P}_m^{-1}\hat{\mathbf{x}}_m + \mathbf{P}_1^{-1}\hat{\mathbf{x}}_1 + \mathbf{P}_2^{-1}\hat{\mathbf{x}}_2 + \ldots + \mathbf{P}_s^{-1}\hat{\mathbf{x}}_s \end{aligned} \tag{18}$$

where $\mathbf{P}_j^{-1}$ is the information matrix for the j^{th} filter at time t_i^- or t_i^+ and it is assumed that all cross-correlation terms among filters are zero as mentioned above. Now, starting with the full solution matrix, this solution can be divided so that the local filters and the master filter each receive share fractions β_j of the total information

$$\begin{aligned} \mathbf{P}_f^{-1} &= \mathbf{P}_m^{-1} + \mathbf{P}_1^{-1} + \mathbf{P}_2^{-1} + \ldots + \mathbf{P}_s^{-1} \\ &= \mathbf{P}_f^{-1}\beta_m + \mathbf{P}_f^{-1}\beta_1 + \mathbf{P}_f^{-1}\beta_2 + \ldots + \mathbf{P}_f^{-1}\beta_s \\ \text{and} \qquad & \mathbf{P}_j^{-1} = \mathbf{P}_f^{-1}\beta_j \text{ or } \mathbf{P}_j = \mathbf{P}_f\beta_j^{-1} \end{aligned} \tag{19}$$

where the "conservation of information" principle [8–10,37] requires that the share-fraction values sum to unity

$$\sum_{j=1}^{m,s}\beta_j = \beta_m + \sum_{k=1}^{s}\beta_k = 1 \tag{20}$$

so that the LF's and MF fractions can be recombined to yield the total correct solution per Equation (18). Note that the index k includes the LF's ($k = 1, 2, ..., s$) and the index j includes both the LF's and the MF ($j = m; 1, 2, ..., s$).

Next, consider the propagation of the covariance matrix for each filter. This process can be performed by each component filter independently, in parallel, assuming the common process noise information is divided in the same way as the fused solutions; so we have just imposed another

assumed constraint. Thus, the standard covariance propagation equation can be employed

$$\mathbf{P}_j(t_i^-) = \mathbf{\Phi}_j(t_i, t_{i-1})\mathbf{P}_j(t_i^+)\mathbf{\Phi}_j^{\mathrm{T}}(t_i, t_{i-1}) + \mathbf{G}_{\mathrm{d}j}(t_{i-1})\mathbf{Q}_{\mathrm{d}j}(t_{i-1})\mathbf{G}_{\mathrm{d}j}^{\mathrm{T}}(t_{i-1}) \tag{21}$$

where the j ($j = m; 1, 2, ..., s$) subscript represents both the LF's and MF values of $\mathbf{\Phi}$, $\mathbf{G}_{\mathrm{d}}$, and $\mathbf{Q}_{\mathrm{d}}$. Again it is assumed that the LF's and MF are all full-sized; so the state transition matrices $\mathbf{\Phi}_k$ and $\mathbf{\Phi}_m$ are equal to $\mathbf{\Phi}_f$, and the noise distribution matrices $\mathbf{G}_{\mathrm{d}k}$ and $\mathbf{G}_{\mathrm{d}m}$ equal $\mathbf{G}_{\mathrm{d}f}$. The process noise covariance matrices $\mathbf{Q}_{\mathrm{d}k}$ and $\mathbf{Q}_{\mathrm{d}m}$ are governed by the information-sharing rules

$$\begin{aligned} \mathbf{Q}_{\mathrm{d}f}^{-1} &= \mathbf{Q}_{\mathrm{d}m}^{-1} + \mathbf{Q}_{\mathrm{d}1}^{-1} + \mathbf{Q}_{\mathrm{d}2}^{-1} + \ldots + \mathbf{Q}_{\mathrm{d}s}^{-1} \\ \text{and} \qquad \mathbf{Q}_{\mathrm{d}j}^{-1} &= \mathbf{Q}_{\mathrm{d}f}^{-1}\beta_j \ \text{ or } \ \mathbf{Q}_{\mathrm{d}j} = \mathbf{Q}_{\mathrm{d}f}\beta_j^{-1} \end{aligned} \tag{22}$$

Thus, given the components $\mathbf{P}_j$ and $\mathbf{Q}_{\mathrm{d}j}$, obtained by Equations (19) and (22), we can propagate the solution, component-wise, and form the solution $\mathbf{P}_f$

$$\begin{aligned} \sum_{j=1}^{m,s} \mathbf{P}_j^{-1}(t_{i+1}^-) &= \sum_j \left[\mathbf{\Phi}_f \mathbf{P}_f(t_i^+)\beta_j^{-1}\mathbf{\Phi}_f^{\mathrm{T}} + \mathbf{G}_{\mathrm{d}f}\mathbf{Q}_{\mathrm{d}f}\beta_j^{-1}\mathbf{G}_{\mathrm{d}f}^{\mathrm{T}}\right]^{-1} \\ &= \left[\sum_{j=1}^{m,s} \beta_j^{-1}\right] \left[\mathbf{\Phi}_f \mathbf{P}_f(t_i^+)\mathbf{\Phi}_f^{\mathrm{T}} + \mathbf{G}_{\mathrm{d}f}\mathbf{Q}_{\mathrm{d}f}\mathbf{G}_{\mathrm{d}f}^{\mathrm{T}}\right]^{-1} = \mathbf{P}_f^{-1}(t_{i+1}^-) \end{aligned} \tag{23}$$

Finally, for measurement update process, each local filter incorporates discrete measurements $\mathbf{z}_k$ from the k^{th} LF. Measurement information is added to each LF by

$$\begin{aligned} \mathbf{P}_k^{-1+} &= \mathbf{P}_k^{-1-} + \mathbf{H}_k\mathbf{R}_k^{-1}\mathbf{H}_k^{\mathrm{T}} \\ \mathbf{P}_k^{-1+}\hat{\mathbf{x}}_k^+ &= \mathbf{P}_k^{-1-}\hat{\mathbf{x}}_k^- + \mathbf{H}_k\mathbf{R}_k^{-1}\mathbf{z}_k \end{aligned} \tag{24}$$

where the superscript '+' refers to post-measurement values and the '-' refers to pre-measurement values. The fusion algorithm in Equation (18) is then used to find the total solution, i.e., the solution that would be found by a single Kalman filter processing all of the $k = 1, \ldots, s$ measurement sets.

In summary, the federated filter solution will provide the same estimates as that of a single, centralized Kalman filter, and is globally optimal when certain assumptions are satisfied:

1. Each filter employs a single β_j value for all of the full-system states and process noises.

2. Equations (19) and (22) are valid.

3. The information fusion and reset equations (which include Zero, Fusion, and No Reset – see [9, 10]) are performed after every measurement cycle.

4. All filters have the same dimensions and thus the same state-space formulation for the common sensors states (the INS in this discussion).

However, in actual practice, some deviations from these assumptions can be made, with small loss of optimality [8–10,37]. First, the federated filter can be implemented such that the local filters are of minimum size, where each local filter (filter k) contains only the common sensor (INS) states and the states unique to the k^{th} sensor. Thus, the $\mathbf{P}_k$, $\mathbf{\Phi}_k$, $\mathbf{Q}_{dk}$, and $\mathbf{G}_{dk}$ matrices contain only the appropriate common and local k^{th} sensor error state partitions of the full matrices. The β_j fraction values are presumed to apply only to the common states, since only those states are shared among the LF's and the MF. Finally, depending on the DKF implementation mode chosen (see [9, 10] for detailed descriptions of implementation options), the MF may contain only common sensor states, or it may additionally contain some LF sensor error states. In this case, the $\mathbf{P}_m$, $\mathbf{\Phi}_m$, $\mathbf{Q}_{dm}$, and $\mathbf{G}_{dm}$ matrices will contain the appropriate partitions, and the β_m fraction values will be applied accordingly.

Also, there are several reasons for not wanting to perform the fusion/reset process after every measurement update:

- Reduce the computation load caused by the fusion and reset operations.
- Reduce database loads between LF processors and the MF/fusion processor.
- Eliminate the requirement to synchronize the fusion/reset operations with the LF measurement update cycles.
- Prevent corrupting all the LFs and MF with bad data from one of the local filters.

Carlson has shown that the multi-step fusion/reset process can be performed in the same way as the single-step process [8–10, 37]. As in other implementations of standard Kalman filters, the LFs may be designed to perform updates to the MF at reduced rates. However, the MF will ignore some information that could in principle be used. Thus, operating the LF's and MF independently over several time steps is equivalent to ignoring an available measurement at each inner step, a suboptimal (since the MF uses only part of the information potentially available to it) but computationally effective approach.

The DKF may be a viable alternative to the MMAE recommended solution, but was not pursued any further. Nevertheless, it should be considered a strong candidate for further investigation for implementation in this new architecture.

1.3 General Approach

The problem of estimating both parameters and states, effectively, is the focus of this research. The new M^3AE architecture (shown in Figure 2) for adaptive filtering using MMAE and Kalman filtering techniques is proposed, developed, and tested. It simultaneously provides :

- Parameter estimation for sampled-data systems from the MMAE portion
- State estimation for sampled-data systems from the additional single Kalman filter within the M^3AE structure.

The new architecture, M^3AE, provides an enhanced multiple model approach to accurate state and parameter estimation, with better performance than obtainable from previous methods. The elemental filters within the MMAE portion of the algorithm may have different tuning (and even structure and state dimension) than the single filter used to produce accurate state estimates, since this MMAE algorithm is devoted to the goal of accomplishing hypothesis testing effectively. In parallel, there is a state estimator that is devoted to high precision state estimation, as shown in Figure 2. Two previous research examples [69, 70, 78] are used to evaluate the performance of the

M^3AE architecture. The M^3AE outperforms the conventional MMAE and nonadaptive filters used in previous research, as shown in the examples presented in Chapter 4.

For parameter estimation, an MMAE based technique is pursued. The focus is accurate parameter identification, not state estimation; i.e., the parameter estimator is designed and tuned for estimating the current system "operating condition," optimized for distinguishing between possible hypothesized conditions rather than being optimized for state estimation performance. Sheldon's research [69, 70] provides an effective method for choosing the discrete parameter values from the continuous parameter space to be used as a basis for the elemental filters, versus previous *ad hoc* trial and error methods. Moreover, the discretization is optimized for *parameter* estimation in the M^3AE, versus being optimized for state estimation in the conventional MMAE. This discretization yields improved M^3AE performance in both parameter and state estimation. Additionally, Lund's multiple model estimation with Inter-Residual Distance Feedback (IRDF) technique (directed toward model discrimination, not state estimation precision) is used for distinguishing the "correct" current system operating model in an enhanced manner via on-line tuning of filter gains (see Section 2.3). IRDF can be used much more effectively in the M^3AE architecture than in the conventional MMAE architecture for which it was originally formulated, as will be shown in this research.

Additionally, given an accurate parameter estimate, it is desired to estimate the system states accurately. Thus, the single Kalman filter state estimator within the M^3AE must be designed and tuned for good state estimation, once told the unknown parameters $\hat{\mathbf{a}}$, of the system, by the MMAE portion of the M^3AE architecture.

Finally, given an M^3AE-based design, an evaluation tool is essential for predicting performance as well as for filter tuning. Therefore, a new approximate covariance analysis tool is developed and implemented for the M^3AE architecture. Figure 3 shows how this tool could be used during the design process.

Two examples demonstrate the M^3AE's performance improvement over that of conventional MMAE's from previously published results. The first example involves a simple second-order mechanical translational system, in which the system's undamped natural frequency is the uncertain parameter. The second example involves a 13-state nonlinear Global Positioning System/Inertial Navigation System (GPS/INS) integration, in which the variance of the measurement noise affecting the GPS outputs is the uncertain parameter.

1.4 Summary

Chapter 2 presents conventional MMAE theory in more detail and sets the foundation for the analytical development of the M^3AE architecture presented in Chapter 3. Chapter 2 also summarizes various adaptations to the basic MMAE technique including: Sheldon's optimal parameter discretization technique [69, 70], Lund's IRDF MMAE [35], moving-bank MMAE [20, 37, 46, 68], and hierarchical MMAE [50, 73].

An MMAE-based architecture provides the foundation upon which the M^3AE is laid. Chapter 3 develops the explicit recursions related to the new M^3AE architecture which solves the problem of estimating states and parameters simultaneously. It provides the analytical development associated with the new M^3AE architecture. Additionally, an approximate covariance analysis tool is developed for the M^3AE architecture which gives the designer the ability to analyze and predict system performance *before* actually implementing the M^3AE or conducting a full-scale Monte Carlo analysis of its capabilities for a given application.

The MMAE-based parameter estimator in M^3AE architecture is designed using Sheldon's optimal parameter discretization technique which assumes steady state, constant-gain Kalman filters [69]. However, in actual practice, many systems never achieve steady state since they are nonlinear and possibly unstable or astable. Therefore, Chapter 3 also presents an extension to Sheldon's opti-

mal parameter discretization technique for unstable or astable, or time-varying and possibly nonlinear, problems using a finite horizon assumption and constrained optimization techniques to generate an approximate solution to the parameter discretization problem.

The MMAE-based parameter estimator is further enhanced through Lund's IRDF method applied to sampled-data measurements case. Chapter 3 develops the discrete-time IRDF, which will be shown to be more useful when applied to the M^3AE architecture than to the standard MMAE, for which it was developed originally.

To demonstrate the M^3AE's performance improvement over previously published results involving conventional MMAE's, two examples are developed and their results presented in Chapter 4. The first example involves a simple second-order mechanical translational system, in which the system's undamped natural frequency is the uncertain parameter. The second example involves a 13-state nonlinear integrated GPS/INS system, in which measurement noise affecting the GPS outputs is the uncertain parameter. The results demonstrate application of the theory, the viability of performance predictions, and the actual performance achievable with an M^3AE compared to conventional MMAE and other methods. Finally, Chapter 5 presents the conclusions derived from this research and proposes areas for future research.

Chapter 2 - MMAE Development

This chapter describes the MMAE in greater detail to set the foundation for development of the M^3AE architecture in Chapter 3. The MMAE was originally proposed by Magill to provide accurate state estimation under parameter variations [36]. MMAE techniques have been investigated since the early seventies with a significant increase in research occurring subsequent to the development of microprocessors and distributed computation [2,3,14,22,36,44,46]. This section will present the fundamentals of the MMAE algorithm in detail and various techniques that have been researched to enhance the performance of the MMAE concept, including Sheldon's optimal discretization of a continuous parameter space for the basis of MMAE elemental filter designs, Lund's inter-residual distance feedback technique for enhanced parameter estimation, Moving-Bank MMAE, and hierarchical structures.

2.1 MMAE Fundamental Development

The basic concept of MMAE was presented in Section 1.2.3 under the assumptions listed below [44,46,75]:

- The sampled-data system is adequately represented by linear stochastic state models for a given parameter vector value, resulting in Gaussian probability density functions, and can be described equivalently by linear stochastic difference equations [44,46]. If nonlinear models are required to describe the system adequately, then extended Kalman filters would replace the linear Kalman filters in the MMAE structure and the associated probability density functions would be approximated as Gaussian. This simplifying assumption enables nonlinear modeling while recognizing the potential suboptimality in assuming Gaussian densities.
- The uncertain parameters to be estimated affect the system matrices or the statistics of the noises entering the system. This assumption covers a very broad class of problems since only the choice of measurement sources and choice of state variables are considered fixed.
- A parameter value typically varies over a continuous range of parameter space. Thus, parameter values will have to be discretized to some level of resolution for feasible implementation. Clearly, poor choices in discrete values for a continuous parameter would result in poor modeling by the MMAE elemental filters and thus poor estimation from the entire MMAE algorithm itself. This results because there might *not* be an elemental filter within the MMAE bank that has a good model of the system's current behavior. Sheldon's research provides a clear method for choosing the discretization required for a design problem [70]. His technique is used in the two examples presented in this research.

Given the assumptions listed above, the development presented in [44, 46, 53, 54] is closely followed. Let $\mathbf{a}$ denote the vector of uncertain parameters in a given linear stochastic state model for a dynamic system (the equations describing this model are presented below). These parameters can influence the matrices defining the structure of the model or depict the statistics of the noises entering it. In order to make simultaneous estimation of states and parameters tractable, it is assumed that $\mathbf{a}$ can take on only one of N_F discrete representative values (discretizing an originally continuous parameter space will be described in detail in Section 2.2).

More explicitly, let the model corresponding to $\mathbf{a}_j$ be described by an "equivalent discrete-time model" ([44, 45]) for a continuous-time system with sampled-data measurements:

$$\mathbf{x}_j(t_i) = \mathbf{\Phi}_j(t_i, t_{i-1})\mathbf{x}_j(t_{i-1}) + \mathbf{B}_{\mathrm{d}j}(t_{i-1})\mathbf{u}(t_{i-1}) + \mathbf{G}_{\mathrm{d}j}(t_{i-1})\mathbf{w}_{\mathrm{d}j}(t_{i-1}) \tag{25}$$

$$\mathbf{z}(t_i) = \mathbf{H}_j(t_i)\mathbf{x}_j(t_i) + \mathbf{v}_j(t_i) \tag{26}$$

where:

$\mathbf{x}$ = n-dimensional system state vector

$\mathbf{\Phi}$ = state transition matrix, the discrete equivalent of the system dynamics matrix

$\mathbf{B}_{\mathrm{d}}$ = discrete equivalent of the system control input matrix

$\mathbf{u}$ = deterministic control input vector

$\mathbf{G}_{\mathrm{d}}$ = discrete equivalent of the noise input matrix

$\mathbf{w}_{\mathrm{d}}$ = discrete-time zero-mean white Gaussian dynamics driving noise vector with covariance $\mathbf{Q}_{\mathrm{d}}(t_i)$ at each t_i

$\mathbf{z}$ = m-dimensional measurement vector

$\mathbf{H}$ = system output matrix

$\mathbf{v}$ = discrete-time zero-mean white Gaussian measurement noise vector with covariance $\mathbf{R}(t_i)$ at t_i; $\mathbf{v}$ is assumed independent of $\mathbf{w}_\mathrm{d}$

Note, $\mathbf{x}(t_0)$ is modeled as Gaussian, with mean $\hat{\mathbf{x}}_{j0}$ and covariance $\mathbf{P}_{j0}$, and is assumed independent of $\mathbf{w}_{\mathrm{d}j}$ and $\mathbf{v}_j$. In general, $\mathbf{a}$ could affect $\mathbf{\Phi}$, $\mathbf{B}_\mathrm{d}$, $\mathbf{G}_\mathrm{d}$, $\mathbf{H}$, $\mathbf{Q}_\mathrm{d}$, $\mathbf{R}$, or even add biases to $\mathbf{w}_\mathrm{d}$ and/or $\mathbf{v}$. The algorithms developed in this research explicitly handle all cases except for added biases, while the examples to follow in Chapter 4 focus on changes in parameters affecting $\mathbf{\Phi}$, $\mathbf{G}_\mathrm{d}$, and $\mathbf{R}$.

Based on this model, the Kalman filter propagation and update equations are given by Equations (27)-(32) with the addition of the subscript j on all variables save $\mathbf{z}$ and $\mathbf{u}$. More explicitly, the propagation equations are

$$\hat{\mathbf{x}}_j(t_i^-) = \mathbf{\Phi}_j(t_i, t_{i-1})\hat{\mathbf{x}}_j(t_{i-1}^+) + \mathbf{B}_{\mathrm{d}j}(t_{i-1})\mathbf{u}(t_{i-1}) \tag{27}$$

$$\mathbf{P}_j(t_i^-) = \mathbf{\Phi}_j(t_i, t_{i-1})\mathbf{P}_j(t_{i-1}^+)\mathbf{\Phi}_j^\mathrm{T}(t_i, t_{i-1}) + \mathbf{G}_{\mathrm{d}j}(t_{i-1})\mathbf{Q}_{\mathrm{d}j}(t_{i-1})\mathbf{G}_{\mathrm{d}j}^\mathrm{T}(t_{i-1}) \tag{28}$$

and the update equations are

$$\mathbf{A}_j(t_i) = \mathbf{H}_j(t_i)\mathbf{P}_j(t_i^-)\mathbf{H}_j^\mathrm{T}(t_i) + \mathbf{R}_j(t_i) \tag{29}$$

$$\mathbf{K}_j(t_i) = \mathbf{P}_j(t_i^-)\mathbf{H}_j^\mathrm{T}(t_i)\mathbf{A}_j^{-1}(t_i) \tag{30}$$

$$\hat{\mathbf{x}}_j(t_i^+) = \hat{\mathbf{x}}_j(t_i^-) + \mathbf{K}_j(t_i)[\mathbf{z}(t_i) - \mathbf{H}_j(t_i)\hat{\mathbf{x}}_j(t_i^-)] \tag{31}$$

$$\mathbf{P}_j(t_i^+) = \mathbf{P}_j(t_i^-) - \mathbf{K}_j(t_i)\mathbf{H}_j(t_i)\mathbf{P}_j(t_i^-) \tag{32}$$

Next, define the hypothesis conditional probability, $p_j(t_i)$, as the probability that $\mathbf{a}$ assumes the value $\mathbf{a}_j$ (for $j = 1, 2, \ldots, N_F$), conditioned on the observed measurement history to time t_i:

$$p_j(t_i) = \mathrm{Prob}[\mathbf{a} = \mathbf{a}_j | \mathbf{Z}(t_i) = \mathbf{Z}_i] \tag{33}$$

then it can be shown [3,36,43,44] that $p_j(t_i)$ can be evaluated recursively for all j via the iteration

$$p_j(t_i) = \frac{f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\mathbf{a}_j, \mathbf{Z}_{i-1})p_j(t_{i-1})}{\sum_{k=1}^{N_F} f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\mathbf{a}_k, \mathbf{Z}_{i-1})p_k(t_{i-1})} \tag{34}$$

in terms of the previous values of $p_1(t_{i-1}), \ldots, p_{N_F}(t_{i-1})$ and the conditional densities for the current measurement $\mathbf{z}(t_i)$ to be defined explicitly in Equation (37). The initial probability distribution for $p_j(t_0)$, for $j = 1, 2, ..., N_F$, is chosen based on any prior information available; for example, if each model is equally likely, the a priori distribution could be set to $p_j(t_0) = 1/N_F$, for $j = 1, 2, ..., N_F$.

Notationally, the measurement history $\mathbf{Z}(t_i)$ is made up of partitions $\mathbf{z}(t_1), \ldots, \mathbf{z}(t_i)$ that are the measurement vectors available at the sample times $t_1, \ldots, t_i$. Similarly, the realization $\mathbf{Z}_i$ of the measurement history vector has partitions $\mathbf{z}_1, \ldots, \mathbf{z}_i$. Furthermore, the Bayesian minimum mean square error (MMSE) estimate of the state is the probability-weighted average

$$\hat{\mathbf{x}}(t_i^+) = E\{\mathbf{x}(t_i)|\mathbf{Z}(t_i) = \mathbf{Z}_i\} = \sum_{j=1}^{N_F} \hat{\mathbf{x}}_j(t_i^+)p_j(t_i) \tag{35}$$

where $\hat{\mathbf{x}}_j(t_i^+)$ is the state estimate generated by a Kalman filter based on the assumption that the parameter vector equals $\mathbf{a}_j$.

Thus, the MMAE algorithm is composed of a bank of N_F separate Kalman filters, each based on a particular value of the parameter vector $(\mathbf{a}_1, \ldots, \mathbf{a}_{N_F})$, as shown in Figure 5 in Section 1.2.3. When the measurement realization $\mathbf{z}(t_i) = \mathbf{H}(t_i) + \mathbf{v}(t_i)$ becomes available at t_i, the residuals $\mathbf{r}_1(t_i), \ldots, \mathbf{r}_{N_F}(t_i)$ are generated in the N_F filters, as shown in Equation (36):

$$\mathbf{r}_j(t_i) = \mathbf{z}_i - \mathbf{H}_j(t_i)\hat{\mathbf{x}}_j(t_i^-) \tag{36}$$

and used to compute $p_1(t_i), \ldots, p_{N_F}(t_i)$ via Equation (34). Each numerator density function in Equation (34) is given by

$$f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\mathbf{a}_j, \mathbf{Z}_{i-1}) = \beta_j \exp\{\cdot\} \tag{37}$$

where

$$\beta_j = \frac{1}{(2\pi)^{m/2}|\mathbf{A}_j(t_i)|^{1/2}} \tag{38}$$

and the expression in the brackets is

$$\{\cdot\} = \left\{-\frac{1}{2}\mathbf{r}_j^{\mathrm{T}}(t_i)\mathbf{A}_j^{-1}(t_i)\mathbf{r}_j(t_i)\right\} \tag{39}$$

and where m is the measurement vector's dimension and $\mathbf{A}_j(t_i)$ is the residual covariance matrix at time t_i calculated in the j^{th} Kalman filter as in Equation (29). The denominator in Equation (34) is simply the sum of all the computed numerator terms and thus is the scale factor required to ensure that all $p_j(t_i)$ values sum to one.

Note, to allow for changing parameter values, some ad hoc compensations suggested by Maybeck are [44]:

1. Put artificial lower bounds on the hypothesis conditional probabilities to avoid estimator lockup. Otherwise once the true parameter is identified, the algorithm can become locked to a single filter output, and mismatched filter estimates can drift significantly from true state values.

2. Add pseudonoise of appropriate strength to the filter models – this causes each Kalman filter to generate state estimates sufficiently close to the true states to allow adaptation to parameter changes. However, care must be taken to avoid adding too much pseudonoise (to guard against divergence), which could mask differences between good and bad models and thus incapacitate the filter's abilities to identify parameter values correctly. If divergence should occur, then the divergent filters could be restarted by using the nondivergent filters.

3. Match each elemental filter to a time history of parameter values rather than to just one constant

value [11, 44, 51, 82] – but this would require N_F^i elemental filters at sample time t_i, which would be impractical for actual implementation.

4. Use Markov models for parameter variation [11, 44, 55] – but then the designer has to evaluate the probability state transition matrix for that Markov model, and that is usually significantly more difficult than establishing a good lower bound, as suggested in (1) above.

5. Moving-bank MMAE (presented in Section 2.4) [18, 20, 23, 46, 48, 68] is a method which avoids the potentially large number of elemental filters required for an MMAE with parameters that can assume many different values.

One expects the residuals of the Kalman filter based on the best model to have the mean-squared value most in consonance with its own computed $\mathbf{A}_j(t_i)$, whereas "mismatched" filters have larger residuals than anticipated through $\mathbf{A}_j(t_i)$. Therefore, Equations (29)-(37) most heavily weight the filter based on the most likely assumed parameter value. However, the performance of the algorithm depends on there being significant differences in the characteristics of residuals in "correct" versus "mismatched" filters. Each filter should be tuned for best performance when the "true" values of the unknown parameters are identical to its assumed value for these parameters. One should specifically avoid the conservative philosophy of adding considerable dynamics pseudonoise, often used to open the bandwidth of a single Kalman filter to guard against divergence, because this tends to mask the differences between good and bad models. However, if, as a result of such tuning, one of the filters should diverge (which is indicated by residuals with large magnitudes or the term computed in Equation (37) being large), then it can be restarted with the current state estimate from the MMAE as computed from the nondivergent filters. Note, the Interacting Multiple Model (IMM) [5] uses this concept after every sample period by restarting every elemental filter in the bank to $\hat{\mathbf{x}}_{\text{MMAE}}(t_i^+)$, if lower bounds are used rather than Markov models to account for para-

meter variations in time (see the ad hoc compensations presented near the beginning of this section). This concept may provide good results for state estimation, but for parameter identification, a subtle change in parameter value may not be detected.

Additionally, given the conditional probability distribution, either the Bayesian or maximum a posteriori (MAP) methods are used to calculate state and parameter estimates. The Bayesian method produces a state estimate which is the conditional mean of the state vector $\mathbf{x}$. The conditional mean $\mathbf{x}$ is:

$$\hat{\mathbf{x}}_{\text{MMAE}}(t_i^+) = \sum_{j=1}^{N_F} \hat{\mathbf{x}}_j(t_i^+) p_j(t_i) \tag{40}$$

where $\hat{\mathbf{x}}_{\text{MMAE}}(t_i^+)$ denotes the estimate at time t_i following the measurement update. Similarly, terms with (t_i^-) denote estimates at time t_i prior to the measurement update. The corresponding parameter estimate (the conditional mean of $\mathbf{a}$), can be generated as

$$\hat{\mathbf{a}}_{\text{MMAE}}(t_i) = \sum_{j=1}^{N_F} \mathbf{a}_j \, p_j(t_i) \tag{41}$$

Alternatively, the MAP method chooses the state (and similarly the parameter estimate) as the values in the model with the highest probability:

$$\hat{\mathbf{x}}_{\text{MAP-MMAE}}(t_i^+) = \hat{\mathbf{x}}_j(t_i^+) \quad \text{for } j \text{ such that } p_j(t_i) = \max_k [p_k(t_i)] \tag{42}$$

and similarly for $\hat{\mathbf{a}}_{\text{MAP-MMAE}}(t_i)$.

Finally, note that the hypothesis conditional probabilities, in Equation (34), at time t_i are functions of the hypothesis conditional probabilities at time t_{i-1}. Due to the recursive nature of the calculations, it is essential that an artificial lower bound be established for $p_j(t_i)$ [22, 44, 46, 53, 54] to allow for parameter variability (or else a Markov model for such variations should be incor-

porated). Without this lower bound, the hypothesis conditional probability of a filter with a totally invalid parameter set for some period of time, $\mathbf{a}_j$, may go to zero and remain zero for all time: once $p_j(t_{i-1})$ reaches zero, $p_j(t_i)$ and all subsequent p_j's would be zero. Thus the j-th filter is essentially removed from the bank. Should the actual system change its characteristics so that the true parameter set a matches $\mathbf{a}_j$ at some future time, $p_j(t_i)$ would remain at zero and the MMAE would produce undesirable results. Even for very small $p_j(t_{i-1})$ (versus actually zero), it would take many sample periods for the probability weight to flow into p_j.

The residual of the j-th filter plays a major role in determining $p_j(t_i)$. The performance of the algorithm is dependent upon a significant difference between the residual characteristics in the "correct" and the "mismatched model" filters. If this criterion is satisfied, then as evident from Equations (34) and (37), the filter with the smallest value of $\left[\mathbf{r}_j^{\mathrm{T}}(t_i)\mathbf{A}_j^{-1}(t_i)\mathbf{r}_j(t_i)\right]$ assumes the largest conditional hypothesis probability. Thus the hypothesis probability algorithm is consistent with the heuristic intuition that the residuals of a well-matched filter should be smaller (relative to the filter's internally computed residual covariance, $\mathbf{A}_j$) than the residuals of a miss-matched filter. However, if in fact, the residuals instead are consistently of the same magnitude, then Equation (34) and (37) result in growth of the p_j associated with the smallest value of $|\mathbf{A}_j|$. The $|\mathbf{A}_j|$ values are independent not only of the residuals, but also of the "correctness" of the N_F models, and so such a result would be totally erroneous. This further emphasizes that it is important not to add too much pseudonoise during tuning, since this tends to mask differences between good and bad models.

Alternate Computation of Probabilities. Whether a Bayesian or MAP form of algorithm is used, the p_j probabilities are computed according to Equation (34), with the first numerator term given by Equation (37). However, it has been frequently noted [2,3,44,50] that the β_j term defined in Equation (38) has nothing to do with the identification of the "correct" parameter value (system operating condition for example), but that all useful information pertaining to "correctness" of pa-

rameter value is confined to the quadratic within the exponential, denoted as the likelihood quotient,

$$L_j(t_i) = \mathbf{r}_j^{\mathrm{T}}(t_i)\mathbf{A}_j^{-1}(t_i)\mathbf{r}_j(t_i) \tag{43}$$

As discussed earlier, if it should occur that the likelihood quotients were essentially the same for all j, then the probability calculations of Equation (34) would be driven by those elemental filters with the smallest value of $|\mathbf{A}_j|$. This is an artificial and incorrect bias, since for example, a sensor failure would be modeled by zeroing out a row of the measurement matrix $\mathbf{H}_j(t_i)$, and this might well cause an improper bias towards falsely identifying such sensor failures.

One ad hoc method for remedying this situation is to remove the β_j term from Equation (37). The result would no longer be a proper density function, since the area under it would no longer be unity. However, because of the scaling effect of the denominator of Equation (34), the computed $p_j(t_i)$ values would still sum to one. Removing the artificial predisposition to declaring sensor failures may well justify such an ad hoc change to the probability calculations in some applications. This, however, is *not* pursued in this research.

2.2 Sheldon's Optimal Parameter Discretization

The following section parallels the discussion presented in [77]. For MMAE to be successful, it is critical that the true parameter lie within the bank's range of coverage for adequate parameter estimation, and it is desirable to have it "close" to one of the elemental filters for improved estimation. Thus the level of discretization of a continuous parameter space directly impacts the ability of the filter bank to "surround" the parameter value and its ability to minimize the "distance" between the true parameter value and one assumed by an elemental filter. Before Sheldon's work [69, 70], *ad hoc* methods were used by designers for choosing the level of discretization. For example, one *ad hoc* method for choosing the coarseness of discretization was to vary one parameter of the truth model in one direction of the parameter space at a time, until the elemental filter, based on the

nominal parameter point, yielded unacceptable levels of degraded performance [18, 20, 44, 68, 70]. Sheldon [69] instead chose optimally discretized parameters by minimizing one of three cost functions, depending on design goals of good state estimation, parameter estimation, or state control. Consider the first case, in which the cost functional, C, represents the average value of the mean squared estimation error, where the average is taken as the true parameter ranges over the entire admissible parameter set:

$$C = \frac{\int_{\mathcal{A}} E\{[\mathbf{x}(t) - \hat{\mathbf{x}}(t)]^{\mathrm{T}} \mathbf{W} [\mathbf{x}(t) - \hat{\mathbf{x}}(t)]\} d\mathbf{a}}{\int_{\mathcal{A}} d\mathbf{a}} \tag{44}$$

where

$$\int_{\mathcal{A}} d\mathbf{a} \triangleq \int_{\mathcal{A}_{N_P}} \cdots \int_{\mathcal{A}_2} \int_{\mathcal{A}_1} da_1 da_2 \ldots da_{N_P}$$

where N_P is the number of scalar parameters (the dimension of $\mathbf{a}$), and where $\mathbf{W}$ is a weighting matrix chosen by the designer to place emphasis on certain states. Also, $\mathcal{A}$ is the admissible set of parameter values, where each parameter varies over a continuous bounded range. A five-step algorithm was developed by Sheldon which allows the designer to approximate and minimize the cost functional numerically [69]. This procedure is accomplished prior to the real-time implementation of the MMAE and provides the designer with the optimal choice for the discretization level. The basic question being addressed is, "If allowed N_F discretized points (where N_F is preselected) in the parameter space, where should they be placed in order to yield optimal MMAE performance relative to the chosen cost function C?" In general, the three cost functions described above will yield different discretizations; so the designer must determine which cost criterion is most pertinent to the problem at hand.

In the M^{3}AE architecture, the MMAE's purpose is parameter estimation, therefore, the cost functional for parameter estimation [69],

$$C = \frac{\int_{\mathcal{A}} E\{[\mathbf{a}(t) - \hat{\mathbf{a}}(t)]^{\mathrm{T}} \mathbf{W} [\mathbf{a}(t) - \hat{\mathbf{a}}(t)]\} d\mathbf{a}}{\int_{\mathcal{A}} d\mathbf{a}} \tag{45}$$

is required. The following summary is taken from Sheldon's dissertation [69] and highlights his assumptions, concepts, and presents his design algorithm used in this research.

Some assumptions, besides those stated earlier for the MMAE development, are required to bound the problem and form the basis for the mathematical development.

- The structure of the plant is known except for an N_P-vector of parameters, which is assumed to be a member of an infinite set.
- The set is bounded and connected so that it makes sense to integrate over the set numerically, in order to find the mean square error.
- There are a finite number of filters available for the estimation.
- *Pseudo*-probabilities are calculated by Equation (34),

$$p_j(t_i) = \frac{f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\mathbf{a}_j, \mathbf{Z}_{i-1}) p_j(t_{i-1})}{\sum_{k=1}^{N_F} f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\mathbf{a}_k, \mathbf{Z}_{i-1}) p_k(t_{i-1})}.$$

  for $j = 1, 2, ..., N_F$. The probabilities are referred to as pseudo-probabilities since it is impossible to have an infinite bank of filters, hence the MMAE algorithms might be calculating the probability that each filter is the correct one, when in fact none of them are.
- The Bayesian estimate formulations in Equations (40) and (41),

$$\hat{\mathbf{x}}_{\mathrm{MMAE}}(t_i^+) = \sum_{j=1}^{N_F} \hat{\mathbf{x}}_j(t_i^+) p_j(t_i)$$

$$\hat{\mathbf{a}}_{\mathrm{MMAE}}(t_i) \triangleq \sum_{j=1}^{N_F} \mathbf{a}_j \, p_j(t_i)$$

  are used to calculate the estimate.
- The MMAE system converges to the model closest, in the Baram-sense (in the information measure of Baram [4]), to the true parameter value with probability 1.
- The development assumes steady state, constant-gain filters (i.e., the $\mathbf{K}_j$'s do not vary with time). However, Sheldon points out that the filters in the MMAE do not have to be steady state Kalman filters. But for this development to apply, the filters must have the same structure as a Kalman filter. Therefore, any other method to calculate a constant gain that produces a stable estimator may be used [70]. For example, in the second problem researched in Chapter 4 (an unstable,

nonlinear, integrated GPS/INS problem using extended Kalman filters), a nominal trajectory point is chosen as the basis for defining the system matrices for determining a "*pseudo*"-constant gain value. Once the gains are calculated, the Sheldon algorithm may be applied using a finite horizon assumption to generate an approximate solution, since the system never achieves steady state. The finite horizon is chosen by the designer, who must choose how much of a finite period of time is of real concern, physically, in a given problem. This approach will be discussed further in Chapter 3 with the results of this technique presented in Chapter 4.

Given these assumptions, the goal is to minimize the cost functional

$$C = \frac{\int_{\mathcal{A}} E\{[\mathbf{a}(t) - \hat{\mathbf{a}}(t)]^{\mathrm{T}} \mathbf{W} [\mathbf{a}(t) - \hat{\mathbf{a}}(t)]\} d\mathbf{a}}{\int_{\mathcal{A}} d\mathbf{a}} \tag{46}$$

Since the admissable parameter set is assumed constant for a given problem, only the numerator of Equation (46),

$$\int_{\mathcal{A}} E\{[\mathbf{a}(t) - \hat{\mathbf{a}}(t)]^{\mathrm{T}} \mathbf{W} [\mathbf{a}(t) - \hat{\mathbf{a}}(t)]\} d\mathbf{a} = \int_{\mathcal{A}} tr\left(\mathbf{W} E\{[\mathbf{a}(t) - \hat{\mathbf{a}}(t)][\mathbf{a}(t) - \hat{\mathbf{a}}(t)]^{\mathrm{T}}\}\right) d\mathbf{a} \tag{47}$$

needs to be minimized. This simply entails a sensitivity analysis into the effect of parameter variations in the elemental filters of the MMAE.

The error autocorrelation equations are derived from the standard Kalman filter equations (25) and (26) based on the design model hypothesizing the parameter value $\mathbf{a}_j$,

$$\mathbf{x}_j(t_i) = \mathbf{\Phi}_j(t_i, t_{i-1}) \mathbf{x}_j(t_{i-1}) + \mathbf{G}_{\mathrm{d}j}(t_{i-1}) \mathbf{w}_{\mathrm{d}j}(t_{i-1}) \tag{48}$$

$$\mathbf{z}(t_i) = \mathbf{H}_j(t_i) \mathbf{x}_j(t_i) + \mathbf{v}_j(t_i) \tag{49}$$

which varies from the actual truth system based on the true parameter $\mathbf{a}_{\mathrm{T}}$:

$$\mathbf{x}_{\mathrm{T}}(t_i) = \mathbf{\Phi}_{\mathrm{T}}(t_i, t_{i-1}) \mathbf{x}_{\mathrm{T}}(t_{i-1}) + \mathbf{G}_{\mathrm{dT}}(t_{i-1}) \mathbf{w}_{\mathrm{dT}}(t_{i-1}) \tag{50}$$

$$\mathbf{z}_{\mathrm{T}}(t_i) = \mathbf{H}_{\mathrm{T}}(t_i) \mathbf{x}_{\mathrm{T}}(t_i) + \mathbf{v}_{\mathrm{T}}(t_i) \tag{51}$$

Given these standard equations, the estimation error vector is given by

$$\tilde{\mathbf{x}}_j(t_i^-) = \hat{\mathbf{x}}_j(t_i^-) - \mathbf{T}\mathbf{x}_{\mathrm{T}}(t_i^-) \tag{52}$$

where $\mathbf{T}$ denotes a transformation matrix from the true state space to the model state space, which allows reduced order filter designs.

The appropriate MMAE filter selections are made using the one-step prediction model of the state estimate $\hat{\mathbf{x}}_j(t_i^-)$ [4]. Sheldon then derived the one-step prediction model from the standard Kalman filter equations in terms of the estimation error vector $\tilde{\mathbf{x}}_j(t_i^-)$, to produce the autocorrelation matrix equation (the time arguments, (t_i), are dropped for convenience) [70]:

$$E\left\{\begin{bmatrix} \tilde{\mathbf{x}}_j^- \\ \mathbf{x}_{\mathrm{T}}^- \end{bmatrix}\begin{bmatrix} \tilde{\mathbf{x}}_j^{-\mathrm{T}} & \mathbf{x}_{\mathrm{T}}^{-\mathrm{T}} \end{bmatrix}\right\} = \mathbf{\Gamma}_j(t_i) = \mathbf{\Upsilon}\mathbf{\Gamma}_j(t_{i-1})\mathbf{\Upsilon}^{\mathrm{T}} + \mathbf{G}_0\mathbf{Q}_0\mathbf{G}_0^{\mathrm{T}} \tag{53}$$

where

$$\mathbf{\Upsilon} = \begin{bmatrix} \mathbf{\Phi}_j(\mathbf{I} - \mathbf{K}_j\mathbf{H}_j) & \mathbf{T}\triangle\mathbf{\Phi} - \mathbf{\Phi}_j\mathbf{K}_j\triangle\mathbf{H} \\ \mathbf{0} & \mathbf{\Phi}_{\mathrm{T}} \end{bmatrix} \tag{54}$$

$$\mathbf{G}_0 = \begin{bmatrix} -\mathbf{T}\mathbf{G}_{\mathrm{dT}} & \mathbf{\Phi}_j\mathbf{K}_j \\ \mathbf{G}_{\mathrm{dT}} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{Q}_0 = \begin{bmatrix} \mathbf{Q}_{\mathrm{dT}} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathrm{T}} \end{bmatrix}$$

and

$$\mathbf{T}\triangle\mathbf{\Phi} \triangleq \mathbf{\Phi}_j\mathbf{T} - \mathbf{T}\mathbf{\Phi}_{\mathrm{T}}$$

$$\triangle\mathbf{H} \triangleq \mathbf{H}_j\mathbf{T} - \mathbf{H}_{\mathrm{T}}$$

If $\mathbf{\Upsilon}$ is a contraction, as $t_i \longrightarrow \infty$, $\mathbf{\Gamma}_j(t_i)$ approaches a constant matrix which is the steady state error prediction autocorrelation, and is denoted $\mathbf{\Gamma}_j^{\infty}$. The upper left partition of $\mathbf{\Gamma}_j(t_i)$, $E\left\{\left[\tilde{\mathbf{x}}_j^-\right]\left[\tilde{\mathbf{x}}_j^-\right]^{\mathrm{T}}\right\}$, is $\mathbf{\Psi}_j^-$, the autocorrelation of the estimator prediction error for the jth filter, and $E\left\{\left[\tilde{\mathbf{x}}_j^-\right]\mathbf{W}\left[\tilde{\mathbf{x}}_j^-\right]^{\mathrm{T}}\right\}$ is calculated by:

$$E\left\{\left[\tilde{\mathbf{x}}_j^-\right]\mathbf{W}\left[\tilde{\mathbf{x}}_j^-\right]^{\mathrm{T}}\right\} = tr\left[\mathbf{W}\mathbf{\Psi}_j^-\right] \tag{55}$$

This value can be calculated at any point in the parameter space once the filter to which the MMAE converges is identified. Given that there are sufficient conditions for the MMAE to converge [4], it will converge to the k^{th} filter governed by [70]:

$$\ell_k = \min \{\ell_j : j = 1, ..., N_F\} \tag{56}$$

where ℓ_j is defined as the proximity of the j^{th} filter generated by [4] as:

$$\ell_j \triangleq \log_e |\mathbf{A}_j| + tr \left\{ [\mathbf{A}_j]^{-1} \left[\begin{bmatrix} \mathbf{H}_j & \triangle\mathbf{H} \end{bmatrix} \mathbf{\Gamma}_j^{\infty} \begin{bmatrix} \mathbf{H}_j & \triangle\mathbf{H} \end{bmatrix}^{\mathrm{T}} + \mathbf{R}_{\mathrm{T}} \right] \right\} \tag{57}$$

given

$$\mathbf{A}_j = \mathbf{H}_j \mathbf{P}_j^{-} \mathbf{H}_j^{\mathrm{T}} + \mathbf{R}_j \tag{58}$$

Sheldon developed a five-step algorithm allowing the designer to approximate and minimize the cost functional given in Equation (44) or (46) numerically [69]:

1. Start by describing the system in terms of the parameter vector $\mathbf{a}$, specifying the structure of the truth system and the filters to be implemented in the estimator.

2. Choose N_F, the number of filters to be implemented in the estimator.

3. Choose a representative parameter set to begin the minimization. The set should be chosen based on which cost functional is being minimized:
   - For the cost function that penalizes *state estimator* errors, a reasonable choice can be obtained by plotting the optimal estimation error autocorrelation, Equation (55), for a coarse discretization of the parameter space assuming $\mathbf{a}_j = \mathbf{a}_{\mathrm{T}}$ at each point. The larger $tr\left[\mathbf{W}\mathbf{\Psi}^{-}\right]$ is, the closer the representative parameters should be.
   - For the cost function that penalizes *parameter estimator* errors, a reasonable choice may be made by equally dividing the admissible parameter region into N_F equal intervals and then choosing the midpoint of each interval as the representative parameter set.

4. Use a numerical integration technique to evaluate Equation (44) or (46) for any given choice of a representative parameter set. The functional evaluations required at each interval are calculated with Equation (53). The information required to set up Equation (53) is based on the proper filter selection per Equation (57). Note, for the cost functional pertaining to the parameter

estimator, the value of $\hat{\mathbf{a}}$ needed in Equation (46) is the value of $\mathbf{a}$ forming the basis of the filter selected at the evaluation point.

5. Step 4 generates a numerical approximation to the value of $C(\mathbf{a})$ for a prespecified choice of parameter vector $\mathbf{a}$. The problem becomes a vector minimization problem, and a vector minimization technique can be used to minimize $C(\mathbf{a})$ using the procedure in step 4 to evaluate the functional for each parameter vector.

To avoid estimator lockup, Sheldon extended his work to account for the design practice of placing lower bounds on the elemental filter probabilities, p_j. The lower bounds, $p_{\min}$, are used to ensure that the system is adaptable to true parameter value changes [70]. The parameter estimate is then given by

$$\begin{aligned} \hat{\mathbf{a}}(t_i) &= \sum_{j=1}^{N_F} \mathbf{a}_j \, p_j(t_i) \\ &= \left\{ [1 - (N_F \cdot p_{\min})] \, \mathbf{a}^{sel}(t_i) \right\} + \sum_{j=1}^{N_F} \mathbf{a}_j(t_i^+) p_{\min} \end{aligned} \tag{59}$$

where $\mathbf{a}^{sel}$ indicates the filter-assumed parameter value selected via Equation (56), i.e., the k^{th} filter to which the MMAE converged in the "Baram sense" [4]. The cost function in Equation (46) is now solved using $\hat{\mathbf{a}}(t_i)$ from Equation (59).

This algorithm was implemented using MATLAB [40] for the examples given in Chapter 4. MATLAB code was developed and written to generate the required parameter sets for each example. A modified Simpson's rule [77] was the numerical integration technique used, and the vector minimization was accomplished using MATLAB's constrained optimization techniques which are based on Sequential Quadratic Programming (SQP) [40]. This will be discussed further in Chapter 3.

2.3 Inter-Residual Distance Feedback (IRDF)

The common assumption used when employing MMAE techniques, is that the parameters take on only a finite number of different values. An MMAE's successful operation depends on the distinguishability of the models used and the tuning of the filters based on the parameters chosen. There must be significant differences in the characteristics of the residual in the "correct" versus "mismatched" filters. Each filter should be tuned for best performance when the "true" values of the unknown parameters are identical to its assumed value for these parameters. When Kalman filters are used in the bank, conservative tuning should be avoided to prevent the residuals from becoming too *close* together and affecting the discrimination property of the filter bank. For fast and reliable discrimination, the residuals should be as distinct as possible [35,44].

Lund has proposed a modification to the MMAE concept and has successfully demonstrated it, by simulation, for a second order single-input single-output (SISO) system [34,35]. The method, Inter-Residual Distance Feedback (IRDF), provides for on-line modification of the elemental filters for the purpose of maintaining the discrimination property of the MMAE filter bank. The method modifies the elemental filters to keep the predicted measurements from becoming too close in "some sense", thus affecting the distinguishability of the elemental filters and thereby the properties of the MMAE algorithm. The elemental filters are modified by detuning the filters through modulation of either the dynamic driving noise covariance $\mathbf{Q}_j$ or the new information $\mathbf{K}_j\mathbf{r}_j$ directly. Recall that $\mathbf{K}_j$ is the j^{th} elemental filter gain matrix and $\mathbf{r}_j$ is the residual vector of the j^{th} elemental filter. Modulation is governed by a scalar quantity computed from a distance measure between the residuals.

Additionally, Lund stresses the trade-off problem in the MMAE concept between that of discrimination versus state tracking. Tracking is the ability of the filter to predict the state $\mathbf{x}(t_i)$ and output $\mathbf{z}_i$ given $\mathbf{Z}_{i-1}$. Thus, the trade-off problem is the desire for good tracking capabilities when

the true system equals one of the models, versus the desire for the residuals of the various elemental filters to be distant from each other, which enables fast and reliable model discrimination. The side of the trade-off which is favored depends on how strongly each filter updates its state estimates from the measurements. Specifically, if small Kalman filter gains are used to de-emphasize the measurement information, then the residuals of the various elemental filters will tend to be more distant from each other. However, this trade-off doesn't impact the M^{3}AE architecture as much as a conventional MMAE since the MMAE portion of the M^{3}AE architecture is not responsible for generating the state estimates. Moreover, a conventional fixed-bank MMAE with parameter space discretized (by Sheldon's methodology from Section 2.2) for $\hat{\mathbf{a}}$ performance versus $\hat{\mathbf{x}}$ performance, enables the designer to exploit the enhanced discrimination information and send a "good" parameter estimate, $\hat{\mathbf{a}}$, to the single Kalman filter within the M^{3}AE that is itself designed for state estimation. Therefore, this combined enhancement could show substantial improvement over conventional MMAE's.

Following Lund's [34, 35] development, let the model for each elemental filter be denoted by M_j from the set $\mathcal{M} = \{M_1, M_2, \ldots, M_{N_F}\}$, and within the limitations of reduced-order modeling, M_j describes the true system, S^*, when operating in mode j. As in Equation (33), the MMAE calculates the probability of each model based on the discrete-time measurement history

$$p_j(t_i) = \text{Prob}[S^* = M_j | \mathbf{Z}(t_i) = \mathbf{Z}_i] \tag{60}$$

For a properly designed MMAE, it is expected that, when $S^* = M_j$,

$$L_j(t_i) << L_k(t_i), \ \forall k \neq j \tag{61}$$

given

$$L_j(t_i) = \mathbf{r}_j^{\mathrm{T}}(t_i)\mathbf{A}_j^{-1}(t_i)\mathbf{r}_j(t_i)$$

which should denote regular behavior of the residuals. Furthermore, $p_j(t_i)$ should increase towards one, and the mismatched filters' probabilities should decrease towards zero, if the condition of Equation (61) continues over several measurements. However, if $S^* \notin \mathcal{M}$ or the filters are not

tuned properly, it is possible that

$$L_1(t_i) \approx L_2(t_i) \approx \ldots \approx L_{N_F}(t_i) \tag{62}$$

and, recalling Equation (37), $p_j(t_i)$ will be directly influenced by $|\mathbf{A}_j(t_i)|$, $j = 1, ..., N_F$. Thus, $p_k(t_i)$ should increase if $|\mathbf{A}_k(t_i)| < |\mathbf{A}_j(t_i)|$, $k \neq j$, while the other $p_j(t_i)$ should decrease. Since, for Kalman filters, $|\mathbf{A}_j(t_i)|$ is not dependent on which model is correct, improper solutions are possible even when using valid models [35, 44]. Hence, the conditions shown in Equation (62) should be avoided through care taken in the development of the algorithm and the tuning of the elemental filters. This highlights the fact that the filters in the bank should not be tuned totally independently [44], and that the distance between residuals should be large enough for inter-residual distinguishability (distant residuals achieve fast and reliable discrimination [35]). Lund defines the following quadratic

$$J_{jk}(t_i) = \mathbf{r}_{jk}^{\mathrm{T}}(t_i)\mathbf{\Gamma}_{jk}\mathbf{r}_{jk}(t_i),\ j \neq k \tag{63}$$

as the squared distance measure of the residual, where the inter-residual difference is $\mathbf{r}_{jk}(t_i) = \mathbf{r}_j(t_i) - \mathbf{r}_k(t_i)$, $j \neq k$, and $\mathbf{r}_j(t_i)$ is the residual in the jth Kalman filter at time t_i ($\mathbf{r}_j(t_i) = \mathbf{z}_i - \mathbf{H}_j(t_i)\hat{\mathbf{x}}_j(t_i^-)$), and $\mathbf{\Gamma}_{jk}$ is a positive definite scaling matrix. Note that $\mathbf{\Gamma}_{jk}$ does not have to be diagonal but is often chosen as such for simplicity. The main principle of this method is to keep $J_{jk}(t_i)$ above some specified limit, $J_{jk}^0(t_i)$, by adjusting filter gains. Note that, if the condition in Equation (62) is present as a result of large dynamics noise strengths, then the filter gains will become large, and

$$\mathbf{r}_{jk}(t_i) = \mathbf{H}_k(t_i)\hat{\mathbf{x}}_k(t_i^-) - \mathbf{H}_j(t_i)\hat{\mathbf{x}}_j(t_i^-) \Rightarrow \mathbf{0}, \text{and in turn } J_{jk}(t_i) \Rightarrow 0 \tag{64}$$

which again emphasizes the point that elemental filters should not be tuned independently.

A general way to keep the inter-residual distance measure, $J_{jk}(t_i)$, above some specified limit is to vary the dynamics noise strengths, $\mathbf{Q}_j$, and thus adjust the filter gains. In the general filter continuous-time propagation equation, for the case of continuous-time measurements (Section 3.6

develops the sampled-data measurement version) as actually developed by Lund [35]:

$$\dot{\mathbf{P}}_j(t) = \mathbf{F}_j(t)\mathbf{P}_j(t) + \mathbf{P}_j(t)\mathbf{F}_j^{\mathrm{T}}(t) + \mathbf{G}_j(t)\mathbf{Q}_j(t)\mathbf{G}_j^{\mathrm{T}}(t) - \mathbf{P}_j(t)\mathbf{H}_j^{\mathrm{T}}(t)\mathbf{R}_j^{-1}(t)\mathbf{H}_j(t)\mathbf{P}_j(t) \quad (65)$$

$\mathbf{Q}_j$ would be replaced by

$$\mathbf{Q}'_j(t) = \eta(t)\mathbf{Q}_j, \; j = 1, \ldots, N_F \quad (66)$$

where the modulating parameter, $\eta(t) \in [\eta_{\min}, 1.0]$. This is a reasonable constraint since the goal is to detune the elemental filters from their nominal operation so as to enhance residual distinguishability, because during nominal operation, the tracking capability of each elemental filter is assumed to be adequate. A value $\eta(t) > 1$ would decrease distinguishability and thus hamper parameter estimation. Finally, the lower bound $\eta_{\min} \geq 0$, must be chosen such that the system maintains stability and the modulation would prevent the tracking capability of the most valid filter from becoming too poor [35]. Lund suggests using $\eta_{\min} = 0$, provided that the system maintains its stability.

Additionally, in Lund's continuous-time measurement development, the time derivative of the modulating variable $\eta(t)$ is given by

(67)

$$\begin{aligned} \tfrac{d}{dt}\eta(t) &= \xi\left\{J_{jk}(t) - J_{jk}^0(t)\right\}, && \text{Cond 1} \\ &= 0, && \text{Cond 2} \end{aligned} \quad (68)$$

where

$$\begin{aligned} &\text{Cond 1}: \quad \eta(t) \in [\eta_{\min}, 1.0] \\ &\text{Cond 2}: \quad \eta(t) = \eta_{\min} \text{ AND } \xi\left\{J_{jk}(t) - J_{jk}^0(t)\right\} < 0 \qquad \text{OR} \\ &\qquad\quad\; \eta(t) = 1 \text{ AND } \xi\left\{J_{jk}(t) - J_{jk}^0(t)\right\} > 0 \end{aligned}$$

These conditions provide anti-integration-windup [35], of importance whenever considering an algorithm incorporating an integrator, as in the case for Equation (67). The constant, ξ is selected to provide proper attenuation of noise on $\eta(t)$. Lund's *ad hoc* guideline for picking a reasonable value for ξ involves looking at the time constants associated with the MMAE filters during all operating

conditions. Then choose ξ such that $1/\xi$ is larger than the largest filter time constant. This ensures that the adaptation of $\eta(t)$ will be slower than the filter settling time [35].

Furthermore, the lower inter-residual difference limit, J_{jk}^0, is chosen such that adequate event detection is achieved. One *ad hoc* method for determining its value involves investigating actual inter-residual differences, J_{jk}, from a sample run providing desirable performance before IRDF is applied. An initial J_{jk}^0 is then chosen based on analysis of actual J_{jk} values seen throughout the sample run (the mean value of J_{jk} is a "good" first choice for J_{jk}^0).

Once initial values of $\eta_{\min}$, ξ, and J_{jk}^0 are chosen, simulations must be conducted to determine if the system meets desired performance requirements. If not, these values must be "tuned" based on insights gained during analysis of the simulation results, until desired performance is achieved.

Decreasing the strengths decreases the Kalman filter gains and in turn increases the value of the distance measure given by Equation (63). Thus, smaller strengths lead to more distinguishability, i.e., a larger $J_{jk}(t)$. For example, if $J_{jk}(t)$ is less than the desired $J_{jk}^0(t)$, Equation (67) will decrease $\eta(t)$ which should, in turn, help drive $J_{jk}(t)$ to the desired value for increased distinguishability. However, a drawback in this technique occurs since the modulated process noise strength, $\mathbf{Q}_j'(t)$, is a function of time and its value is only known in real-time operation, and thus filter gains cannot be precomputed, even for linear models.

Lund proposed a simplification to this method for linear models by modulating the new information $\mathbf{K}_j(t)\mathbf{r}_j(t)$ by $\eta(t)$ instead of $\mathbf{Q}_j(t)$, as shown by the following equation:

$$\mathbf{K}_j'(t)\mathbf{r}_j(t) = \eta(t)\mathbf{K}_j(t)\mathbf{r}_j(t) \tag{69}$$

The filter gains, $\mathbf{K}_j(t)$, are now precomputable and only the modulation is computed on-line [35]. Also, modulating $\mathbf{K}_j(t)$ versus $\mathbf{Q}_j(t)$ will yield quicker adaptation response since there isn't any wait for the filter state covariance $\mathbf{P}_j(t)$ equations to traverse their transients before the corresponding filter gains are thereby changed. Thus, direct modulation of the new information is a viable

option and Lund recommends that it can be readily accomplished for steady state elemental filters. However, Lund points out that, for extended Kalman filters and higher order filters, there is less benefit to this approach, since filter gains are computed on-line anyhow [35], but the quicker adaptation response is still achieved.

A final concern discussed by Lund is the lack of guaranteed stability for all the elemental filters without placing restrictions on the choice of the modulating parameter, $\eta(t)$. An approach is presented in [35] which requires extensive simulations in the case of extended Kalman filters. When considering the Lund-type MMAE (either for a conventional MMAE or for the MMAE portion of the M^{3}AE), the stability problem is best handled by considering Figure 2 on page 3, in which the MMAE using IRDF could optimize parameter identification and a single Kalman filter (linear or extended) would be used to generate the final state estimate, given $\hat{\mathbf{a}}$. If some of the elemental filters go unstable as a result of the IRDF modulation, then they would be reset by the stable filters as discussed in Section 2.4, where the blended estimate of the stable filters was used as the initial condition for the filters being reset.

IRDF is inherently directed toward model discrimination, not state tracking [20]. Thus, Lund's technique to ensure distant residuals for fast and reliable model discrimination should be an effective method for enhancing parameter estimation. Moreover, in the M^{3}AE structure, unlike the conventional MMAE architecture, this beneficial impact does not have to be traded off against the desire *not* to use artificially low $\mathbf{Q}$ (or $\mathbf{K}$) values when generating a precise state estimate.

2.4 Moving-Bank MMAE

To avoid the potentially large number of elemental filters needed for an MMAE bank, (for example, if there are two uncertain parameters and each can assume 10 possible values, then $10^2 = 100$ separate filters must be implemented, even if the parameters are treated as unknown constants),

the concept of a "moving bank" of fewer filters has been investigated [18,20,46,68]. The moving-bank MMAE is identical to the full-bank estimator discussed previously, except N_F corresponds to the smaller number of elemental filters in the moving bank rather than the total number of possible discrete parameter vector values. In the example above, one might choose the three discrete values of each parameter that most closely surround the estimated value, requiring $3^2 = 9$ separate elemental filters, rather than 100. This is depicted in Figure 7.

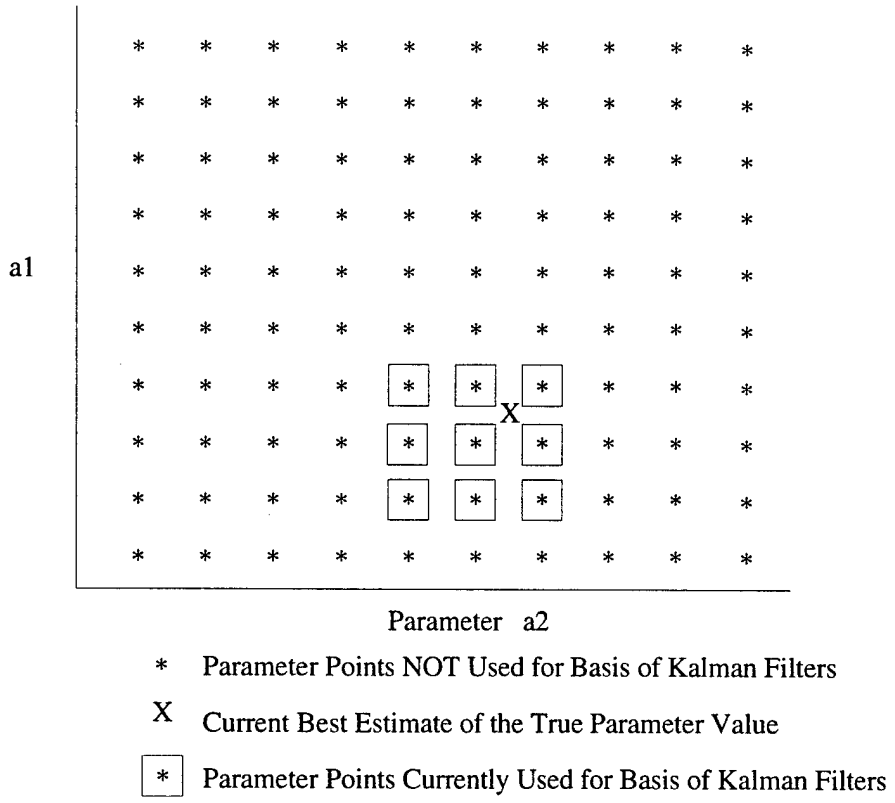


Figure 7. Bank Definition for Moving-Bank MMAE for a Two-Dimensional Parameter Space

Which particular N_F filters are in the bank at a given time can be determined by one of the five decision mechanisms presented later in this section, with the intention of keeping the true value of the parameter in the bounds of the bank. For some of the decision mechanisms, the parameter estimate is used to center the bank of filters. If the parameter estimate is found to move, then the bank of filters will move within the parameter space as shown in Figure 8, thus hopefully tracking the true parameter.

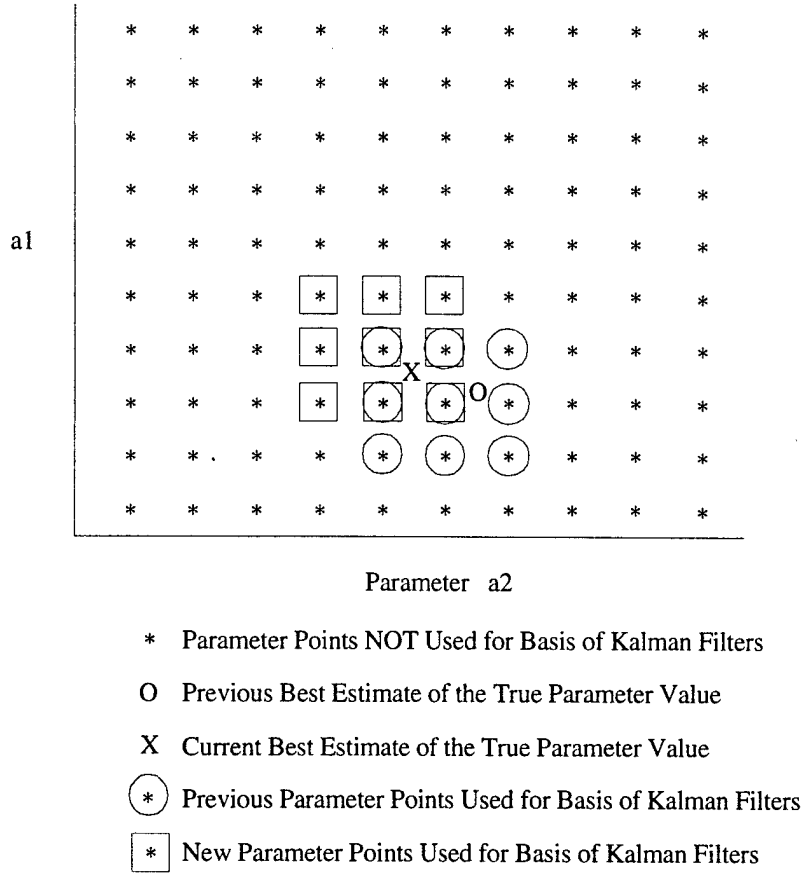


Figure 8. *Moved* Moving-Bank MMAE

For some of the decision mechanisms, it may appear that the true parameter lies outside the bounds of the current bank; so the bank must expand to the coarsest level of discretization to attempt to bring the true parameter value into the bank, as illustrated in Figure 9. Note that moving the bank and expanding the bank are two separate decisions.

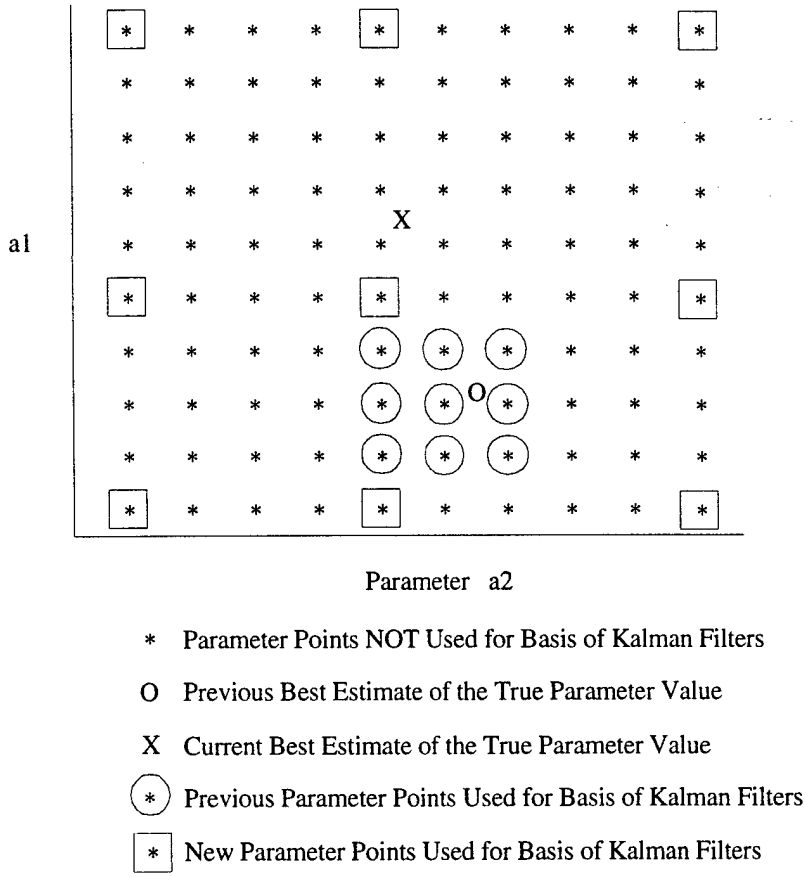


Figure 9. *Expanded* Moving-Bank MMAE

It is critical that the true parameter lie within the bank's range of coverage for adequate parameter estimation, and it is desirable to have it "close" to one of the elemental filters for improved estimation. Thus the level of discretization of a continuous parameter space directly impacts the ability of the filter bank to "surround" the parameter value and its ability to minimize the "dis-

tance" between the true parameter value and one assumed by an elemental filter, as discussed in Section 2.2.

Five decision logics have been investigated by Maybeck and others in previous research efforts for keeping the moving-bank centered about the estimated parameter value [18,20,44,46,68]: 1) residual monitoring, 2) parameter position monitoring, 3) parameter position and velocity monitoring, 4) probability monitoring, and 5) parameter estimation error covariance monitoring [46]. A brief summary of each follows.

Residual Monitoring. Recall the likelihood quotient given by Equation (43):

$$L_j(t_i) = \mathbf{r}_j^{\mathrm{T}}(t_i)\mathbf{A}_j^{-1}(t_i)\mathbf{r}_j(t_i)$$

In the case of scalar measurements, this is the current residual squared, divided by the filter-computed variance for the residual. When the true parameter value does not lie in the center of the moving-bank region, some of the N_F likelihood quotients (the ones corresponding to the $\mathbf{a}_j$ values most distant from the true parameter value) can be expected to exceed a threshold level T, the numerical value of which is set in an *ad hoc* manner during performance evaluations. Thus, a possible detection logic would indicate that the bank should be moved at time t_i if

$$\min[L_1(t_i), L_2(t_i), \ldots, L_j(t_i)] \geq T \tag{70}$$

Moreover, the elemental filter based on $\mathbf{a}_j$ nearest to the true parameter value should have the smallest likelihood quotient, thereby giving an indication of the direction to move the bank. Although this logic would respond effectively to a real need to move the bank, it would also be prone to false alarms induced by single large samples of measurement noise.

Parameter Position Estimate Monitoring. Another means of keeping the true parameter value in the region bracketed by the moving bank is to keep the bank centered (as closely as possible in view of the discrete values that $\mathbf{a}$ is allowed to assume) on the current estimate of the parameter.

This estimate was shown in Equation (41) as:

$$\hat{\mathbf{a}}(t_i) = E\{\mathbf{a}|\mathbf{Z}(t_i) = \mathbf{Z}_i\} = \sum_{j=1}^{N_F} \mathbf{a}_j \, p_j(t_i) \tag{71}$$

If the distance from the parameter value associated with the center of the bank to $\hat{\mathbf{a}}(t_i)$ becomes larger than some chosen threshold, a move of the bank in that direction is indicated. Since $\hat{\mathbf{a}}(t_i)$ depends on a *history* of measurements rather than just the single current measurement, this technique is less prone to the false alarms discussed in the previous method.

Parameter Position and "Velocity" Estimate Monitoring. If the true parameters are slowly varying, past values of $\hat{\mathbf{a}}(t_i)$ can be used to generate an estimate of parameter "velocity". This, along with the current position estimate $\hat{\mathbf{a}}(t_i)$, can be used to compute a predicted parameter position, one sample period into the future. If the distance between the bank center and that projection exceeds some selected threshold, the bank can be moved in anticipation of the true parameter movement. This has the advantage of putting some lead into parameter estimation, but one must be aware of the additional error variance in a prediction (thus a higher level of uncertainty and erratic bank movement) over the variance of the preceding (updated) filter estimate.

Probability Monitoring. The conditional hypothesis probabilities, $p_j(t_i)$, are another indication of the correctness of the parameter values $\mathbf{a}_j$ assumed by the elemental filters of the current bank. If any of these rise above a chosen threshold level, the bank can be moved in the direction of the $\mathbf{a}_j$ associated with the highest $p_j(t_i)$. In this scheme, the bank seeks to center itself on the elemental filter with the highest conditional probability weighting. Again, since $p_j(t_i)$ depends on a history of measurements, this method should not be as sensitive to single bad samples of measurement corruption as is the case under residual monitoring.

In addition to the four decision logics presented above, Li has proposed a "Variable Structure" MMAE [30, 31]. Once a decision has been made to move the bank, the basic concept involves increasing the number of filters in the bank as well as the size of the bank in parameter space,

temporarily, until enough information is processed to determine which elemental filters should be terminated. The advantage to this scheme lies in its potential to reduce the chance of inappropriate and nonsystematic bank motion [30].

Parameter Estimation Error Covariance Monitoring. This concept is discussed last because it has a somewhat different purpose than deciding when and in what direction to move the bank. Here it is also possible to change the size of the bank by altering the discretization level of the parameter space as shown in Figure 9. For example, initial acquisition can be enhanced by choosing the values $\mathbf{a}_1, \ldots, \mathbf{a}_{N_F}$ so that they coarsely encompass all possible parameter values, rather than use a small bank and force it to seek a true parameter value that may well be outside the region of its assumed parameter values. Then, once a "good" parameter estimate has been achieved with this coarse discretization, the size of the bank can be contracted and the smaller bank centered on that good value. To help make such a contraction decision, it would be useful to monitor the parameter estimation error conditional covariance, computable [44] as

$$\begin{aligned} \mathbf{P}_{\mathbf{a}}(t_i) &= E\{[\mathbf{a} - \hat{\mathbf{a}}(t_i)][\mathbf{a} - \hat{\mathbf{a}}(t_i)]^{\mathrm{T}} | \mathbf{Z}(t_i) = \mathbf{Z}_i\} \\ &= \sum_{j=1}^{N_F} [\mathbf{a}_j - \hat{\mathbf{a}}(t_i)][\mathbf{a}_j - \hat{\mathbf{a}}(t_i)]^{\mathrm{T}} \cdot p_j(t_i) \end{aligned} \tag{72}$$

When an appropriately chosen scalar function (norm) of this matrix falls below a selected threshold, the bank can be *contracted* about the parameter estimate.

If the true parameter value were to change (move), such that the true value was now outside the "boundary" of the current bank, then the bank would need to expand. One method of determining whether or not to expand the bank involves the likelihood quotient $L_j(t_i)$, defined in Equation (43) and used in the preceding "Residual Monitoring" section. If *all* the $L_j(t_i)'s$ exceed a predetermined

threshold, indicating *none* of the elemental filters have a good hypothesized a_j value, then the bank should expand [77].

2.5 Hierarchical Structure

Maybeck and his students have researched multiple failures for reconfigurable flight control via MMAE methods [12, 14, 15, 21, 22, 29, 50, 53, 54, 71, 72]. If a multiple model algorithm were based upon all possible single and double failures of K sensors, it would require one elemental filter for the fully functional status, K single-failure elemental filters, and K!/[(K–2)!2!] double-failure filters. To avoid this computational burden, the idea of the "moving-bank" leads to the concept of a hierarchical structure that requires at most only (K+1) elemental filters to be on-line at any given time: the same number used when only single failures are modeled. Figure 10 illustrates such a hierarchical structure.

At "Level Zero", there are K elemental filters specifically designed for one of the single-failure conditions and one configured for the fully functional system (denoted as a_0 in the figure). Upon confirmation that failure "k" (any of the possibilities 1,2,...,K) has occurred, a new MMAE bank is brought on-line from memory at "Level One" and replaces the original "Level Zero" MMAE. It would consist of K+1 elemental filters (where K+1=N_F in the previously used notation): one designed for the k^{th} single-failure condition (denoted a_k), K–1 configured for the double-failure condition of the known k^{th} failure plus one of the remaining possible failures (denoted a_{kl}), and one designed for the fully functional (a_0) system to allow for using future measurements to change the decision that the first failure, had, in fact, occurred.

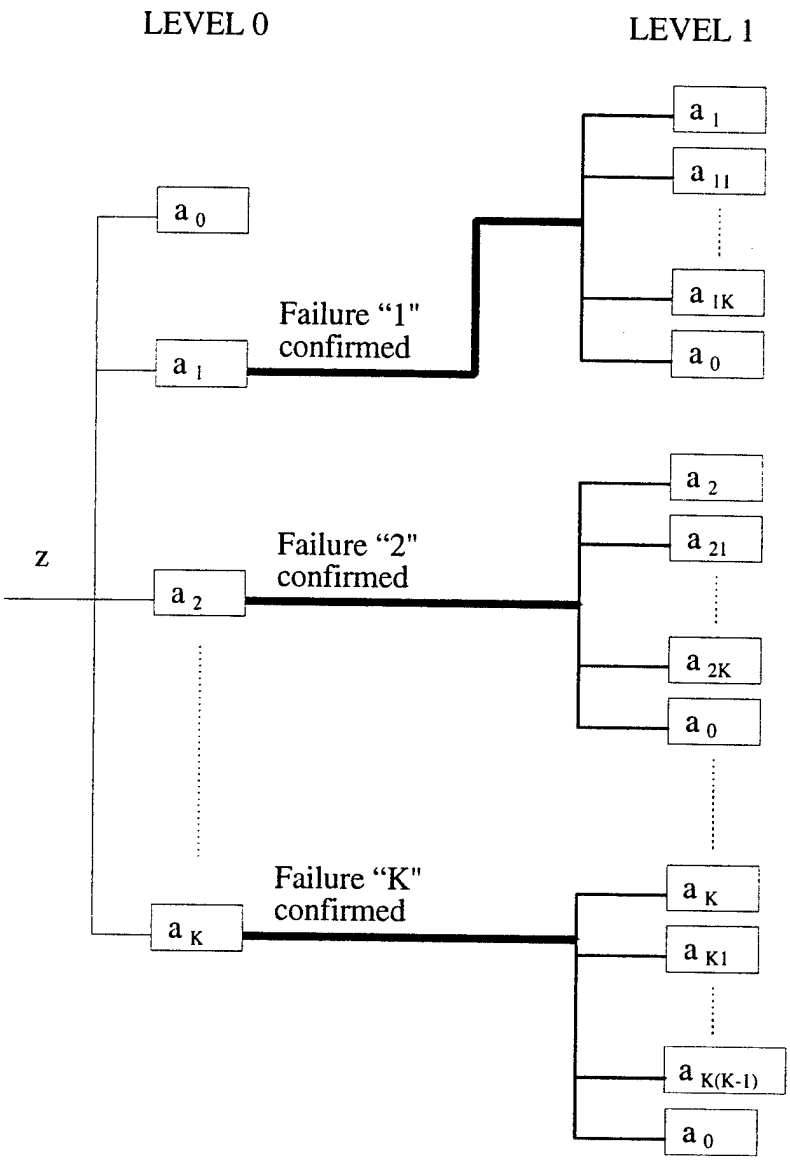


Figure 10. Hierarchical Modeling - Level 0 and Level 1 MMAE Banks

2.6 Summary

The theory and development behind the MMAE and various adaptations have been presented in this chapter. Many options exist for combining the above methods. Of those presented, Sheldon's parameter discretization technique and Lund's IRDF are pursued in this research. With this foundation, the M^3AE formulation can now be developed in Chapter 3. The focus of the MMAE portion of the architecture is on parameter estimation. The state estimation is produced via a separate standard Kalman Filter, fed with the $\hat{\mathbf{a}}$ from the MMAE portion, and tuned for precise state estimation performance.

Chapter 3 - M^{3}AE Development and Performance Analysis

This chapter develops the explicit recursions related to the new M^{3}AE architecture which solves the problem of estimating states and parameters simultaneously. It provides the analytical development associated with the new M^{3}AE architecture shown in Figure 11. Additionally, an approximate covariance analysis tool is developed for the M^{3}AE architecture which gives the designer the ability to analyze and predict system performance before actually implementing the M^{3}AE or running a full-scale Monte Carlo analysis of it.

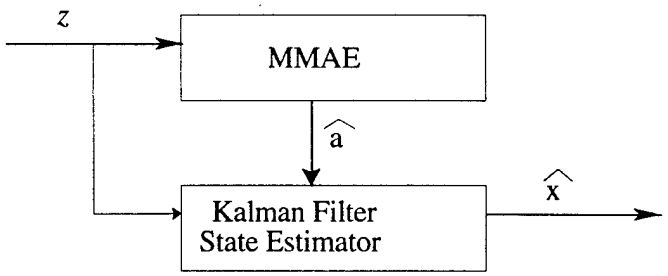


Figure 11. M^{3}AE – MMAE-Based Parameter Estimator and KF-Based State Estimator

A covariance analysis tool (for linear systems driven by white Gaussian noise, which have measurements taken by sensors described by linear models corrupted by white Gaussian noise) has been available for nonadaptive Kalman filter designers. A single run of the covariance analysis generates the entire time history of the covariance of the true estimation errors committed by the Kalman filter. This provides a valuable design tool since initial performance analysis and subsequent filter tuning can be accomplished quickly without any knowledge of the explicit samples of the measurement history [43]. The covariance expressions for the discrete-time model formulation of the Kalman filter based on a "truth model" are:

$$\mathbf{P}(t_i^-) = \mathbf{\Phi}\mathbf{P}(t_{i-1}^+)\mathbf{\Phi}^{\mathrm{T}} + \mathbf{G}_{\mathrm{d}}(t_{i-1})\mathbf{Q}_{\mathrm{d}}(t_{i-1})\mathbf{G}_{\mathrm{d}}^{\mathrm{T}}(t_{i-1}) \tag{73}$$

$$\mathbf{K}(t_i) = \mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i)\left[\mathbf{H}(t_i)\mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i)+\mathbf{R}(t_i)\right]^{-1} \tag{74}$$

$$\mathbf{P}(t_i^+) = \mathbf{P}(t_i^-) - \mathbf{K}(t_i)\mathbf{H}(t_i)\mathbf{P}(t_i^-) \tag{75}$$

Notice, that knowledge of the measurement history is not required to solve these expressions, and as such, they can be completely precomputed. Covariance analysis can also be developed [43] for Kalman filters based on reduced-order, simplified models rather than "truth models," but the focus of this development is on the covariance analysis for filters based on very good models or "truth models."

A similar tool has been available for conventional MMAE's as well. The fundamental conditional covariance expressions of $\mathbf{x}(t_i)$ and $\mathbf{a}(t_i)$ for an MMAE are [44]:

$$\begin{aligned}\mathbf{P}(t_i^+) &= E\left\{\left[\mathbf{x}(t_i)-\hat{\mathbf{x}}(t_i^+)\right]\left[\mathbf{x}(t_i)-\hat{\mathbf{x}}(t_i^+)\right]^{\mathrm{T}}|\mathbf{Z}(t_i)=\mathbf{Z}_i\right\} \\ &= \sum_{j=1}^{N_F} p_j(t_i)\left\{\mathbf{P}_j(t_i^+)+\left[\hat{\mathbf{x}}_j(t_i^+)-\hat{\mathbf{x}}(t_i^+)\right]\left[\hat{\mathbf{x}}_j(t_i^+)-\hat{\mathbf{x}}(t_i^+)\right]^{\mathrm{T}}\right\}\end{aligned} \tag{76}$$

$$\mathbf{P}_{\mathbf{a}}(t_i) = E\{[\mathbf{a}-\hat{\mathbf{a}}(t_i)]\,[\mathbf{a}-\hat{\mathbf{a}}(t_i)]^{\mathrm{T}}\,|\mathbf{Z}(t_i)=\mathbf{Z}_i\} = \sum_{j=1}^{N_F}[\mathbf{a}_j-\hat{\mathbf{a}}(t_i)]\,[\mathbf{a}_j-\hat{\mathbf{a}}(t_i)]^{\mathrm{T}}\,p_j(t_i) \tag{77}$$

where $\mathbf{P}_j(t_i^+)$ is the state error covariance computed by the elemental filter based upon $\mathbf{a}_j$, $\hat{\mathbf{x}}_j(t_i)$ is the corresponding elemental filter state estimate, and $\hat{\mathbf{x}}(t_i^+)$ and $\hat{\mathbf{a}}(t_i)$ are the MMAE estimates of states and parameters, respectively. Notice, however, that in contrast to the single nonadaptive Kalman filter, neither of these expressions can be precomputed since knowledge of $p_j(t_i)$, $\hat{\mathbf{x}}_j(t_i)$, $\hat{\mathbf{x}}(t_i^+)$, and $\hat{\mathbf{a}}(t_i)$ is required. However, a one-run Monte Carlo simulation could provide *representative* measurement history information (thus providing *representative* values for $p_j(t_i)$, $\hat{\mathbf{x}}_j(t_i)$, $\hat{\mathbf{x}}(t_i^+)$, and $\hat{\mathbf{a}}(t_i)$ for all time, conditioned on that measurement time history, $\mathbf{Z}_i$) upon which a co-

variance analysis could be accomplished, thereby avoiding time consuming multi-run Monte Carlo studies in the initial design process (see Section 3.3 for further discussion).

A designer developing a system based on the M^{3}AE architecture could also benefit from such a covariance analysis tool. Therefore, an approximate covariance analysis tool has been developed for the M^{3}AE architecture, as shown previously in Figure 3, and presented in detail in Section 3.4. As with an MMAE design, an M^{3}AE designer may conduct an approximate covariance analysis after only a single Monte Carlo run on *just* the MMAE-based parameter estimator. The single Monte Carlo run provides a *representative* time history of the parameter estimates $\hat{\mathbf{a}}(t_i)$, filter-computed error covariance $\mathbf{P_a}(t_i)$, and the associated elemental filter probabilities $p_j(t_i)$. This information is then provided to the approximate M^{3}AE covariance analysis tool, which produces the required covariance analysis information to conduct a performance or sensitivity study of the M^{3}AE for making design decisions.

The fixed-bank MMAE parameter estimator used in the M^{3}AE structure can be based on an enhanced Sheldon parameter discretization technique which is presented in Section 3.5. Additionally, the fixed-bank MMAE can use Lund's inter-residual distance feedback (IRDF) technique, modified for a discrete-time representation of the system, which is discussed in Section 3.6.

3.1 M^{3}AE Architecture

The M^{3}AE architecture is fairly simple and provides an excellent design option for designers concerned with applications in which accurate parameter and state estimation is the design goal. Figure 11 displays this architecture which offers enhanced design flexibility in optimizing each estimator for its intended purpose. The architecture involves a single KF (designed and tuned for high-fidelity state estimation) that accepts the parameter estimate provided by a fixed-bank MMAE, designed for precise parameter estimation.

Since the goal of the MMAE "block" is accurate parameter identification, the elemental filters are designed using Sheldon's parameter discretization algorithm, specifically using the cost function devoted to minimizing parameter estimation errors. To enhance filter distinguishability further, Lund's IRDF algorithm (whose primary purpose is to keep filter residuals "distant" for the purpose of fast and reliable discrimination of parameter values) is implemented in the fixed-bank MMAE. This is appropriate for an MMAE producing a parameter estimate $\hat{\mathbf{a}}$, as in the M^3AE architecture, but may not be as appropriate for an MMAE designed for state estimation, since the state tracking capability of the MMAE may suffer [35].

3.2 Assumptions

In addition to the underlying assumptions associated with the standard Kalman filter and MMAE theory (see Section 2.1 and [43–45]), the following assumptions pertaining to the M^3AE architecture are emphasized:

1. The structure of the problem is known and modelling techniques have provided a suitable mathematical description in the form of linear difference equations driven by known inputs and white Gaussian noise sequences. Additionally, the measurements taken by sensors are well described by linear models corrupted by white Gaussian noise.

2. To facilitate the future implementation of Assumption 1 in an on-line, digital computer environment; a discrete-time representation of the system dynamics is chosen and represented by the following linear stochastic difference equation associated with the parameter value $\mathbf{a}_j$:

$$\mathbf{x}_j(t_i) = \mathbf{\Phi}_j(t_i, t_{i-1})\mathbf{x}_j(t_{i-1}) + \mathbf{B}_{dj}(t_{i-1})\mathbf{u}(t_{i-1}) + \mathbf{G}_{dj}(t_{i-1})\mathbf{w}_{dj}(t_{i-1}) \tag{78}$$

upon which available discrete-time measurements are modeled by the following linear equation:

$$\mathbf{z}(t_i) = \mathbf{H}_j(t_i)\mathbf{x}_j(t_i) + \mathbf{v}_j(t_i) \tag{79}$$

It is assumed that $\mathbf{w}_d$ and $\mathbf{v}$ are independent, zero-mean, white Gaussian noise processes with covariance kernels:

$$E\{\mathbf{w}_d(t_i)\mathbf{w}_d^T(t_k)\} = \mathbf{Q}_d(t_i)\delta_{ik} \tag{80}$$

$$E\{\mathbf{v}(t_i)\mathbf{v}^T(t_k)\} = \mathbf{R}(t_i)\delta_{ik} \tag{81}$$

where $\mathbf{Q}_d(t_i)$ is positive semi-definite and $\mathbf{R}(t_i)$ is positive definite for all t_i. Additionally, the initial conditions on the states are known, in general, with some uncertainty, and $\mathbf{x}(t_0)$ is also described as a Gaussian random vector independent of $\mathbf{w}_d$ and $\mathbf{v}$, with known mean and covariance:

$$E\{\mathbf{x}(t_0)\} = \hat{\mathbf{x}}_0 \tag{82}$$

$$E\{[\mathbf{x}(t_0) - \hat{\mathbf{x}}_0][\mathbf{x}(t_0) - \hat{\mathbf{x}}_0]^T\} = \mathbf{P}_0 \tag{83}$$

where $\mathbf{P}_0$ is positive semi-definite.

3. This development assumes that the control input is exactly known to the filter.

4. Given that the problem to be solved contains uncertain parameters, $\mathbf{a}$, a complete determination of affected system matrices such as $\mathbf{\Phi}$, $\mathbf{B}_d$, $\mathbf{G}_d$, $\mathbf{H}$, $\mathbf{Q}_d$, and $\mathbf{R}$ is typically not possible because of observability issues. However, for this development, the parameters to be estimated are assumed identifiable.

5. The unknown parameters are assumed to vary much slower than unknown states and in many cases may be time-invariant. Thus, any estimate of uncertain parameters should take advantage of knowledge concerning slow (if any) variability of the parameter from one time sample to the next.

6. Uncertainties in the measurement matrix $\mathbf{H}(t_i)$ are obviously possible, and are considered in the general development to follow. However, for many problems, it is usually not necessary to estimate parameters in $\mathbf{H}(t_i)$. For example, in problems in which the state space model is expressed in physical variables, the measurement matrices are generally known better than $\mathbf{\Phi}(t_i, t_{i-1})$, $\mathbf{B}_{\mathrm{d}}(t_{i-1})$, or $\mathbf{G}_{\mathrm{d}}(t_{i-1})$ [42]. Furthermore, parameter uncertainties in $\mathbf{H}(t_i)$ are difficult to distinguish from uncertainties in $\mathbf{\Phi}(t_i, t_{i-1})$ or $\mathbf{B}_{\mathrm{d}}(t_{i-1})$ or $\mathbf{G}_{\mathrm{d}}(t_{i-1})$, in that alternative state space models with equivalent input/output characteristics may differ from each other in one having uncertain parameters in $\mathbf{H}(t_i)$ (alone or in $\mathbf{\Phi}(t_i, t_{i-1})$, $\mathbf{B}_{\mathrm{d}}(t_{i-1})$, and $\mathbf{G}_{\mathrm{d}}(t_{i-1})$ as well), whereas another may have no uncertainties in $\mathbf{H}(t_i)$. This usually becomes apparent after investigations into measurement, $\mathbf{z}(t_i)$, and control, $\mathbf{u}(t_i)$, time histories. Therefore, it's usually convenient to avoid parameter uncertainties in $\mathbf{H}(t_i)$ [44]. Note that both examples presented in this research do not contain uncertain parameters in $\mathbf{H}(t_i)$ (see Chapter 4).

7. Uncertainties in the input noise matrix $\mathbf{G}_{\mathrm{d}}(t_i)$ are not included since [1, 44] found that they could be equivalently treated as uncertain parameters in the dynamics noise covariance matrix $\mathbf{Q}_{\mathrm{d}}(t_i)$.

8. In the recursive filter implementation accomplished, it is assumed that the uncertain parameter values remain essentially constant between time increments.

9. Complete coverage of the parameter space is typically not possible. Therefore, a range of values is chosen based on an examination of the physics involved in the problem. In turn, the MMAE elemental filter models are based on a limited subset of parameter values chosen from a finite subset of discrete values. This has the benefit of limiting potential computational burdens.

10. The theory is developed based on full-order, linear truth models. Obviously, in actual practice, this is not always possible. Therefore, some deviations from these assumptions occur with an associated loss of "optimality."

3.3 Theory and Evaluation

Given the assumptions presented above, this section develops the theory supporting the new M^{3}AE architecture and the associated conditional covariance expressions. Following the MMAE development of Chapter 2, let $\mathbf{a}$ denote the vector of uncertain parameters in a given model. Assuming a Kalman filter based system, $\mathbf{a}$ is allowed to affect any or all of the $\mathbf{\Phi}$, $\mathbf{B}_d$, $\mathbf{G}_d$, $\mathbf{H}$, $\mathbf{Q}_d$, and $\mathbf{R}$ system matrices.

The parameter estimate $\hat{\mathbf{a}}$ and its error covariance $\mathbf{P_a}$ are provided by the "box" designed for parameter estimation in Figure 1 of Chapter 1 and Figure 11 of this chapter (given the criteria and assumptions stated earlier). The development for this M^{3}AE based architecture assumes an MMAE-based parameter estimator. Thus $\mathbf{a} \in \mathbf{R}^{N_P}$, and then via Sheldon discretization [70], $\mathbf{a}$ is allowed to assume one of N_F (number of filters) values, $\mathbf{a}_j$ for $j = 1, 2, ..., N_F$ within that N_P-dimensional space. Paralleling Maybeck's development in [44], the conditional mean and covariance of $\mathbf{a}$ at time t_i are

$$\begin{aligned}
\hat{\mathbf{a}}(t_i) &\triangleq E\{\mathbf{a}(t_i)|\mathbf{Z}(t_i) = \mathbf{Z}_i\} = \int \boldsymbol{\alpha} f_{\mathbf{a}|\mathbf{Z}(t_i)}(\boldsymbol{\alpha}|\mathbf{Z}_i)\mathrm{d}\boldsymbol{\alpha} \\
&= \int \boldsymbol{\alpha} \left[\sum_{j=1}^{N_F} p_j(t_i)\delta(\boldsymbol{\alpha} - \mathbf{a}_j) \right] \mathrm{d}\boldsymbol{\alpha} \\
&= \sum_{j=1}^{N_F} \mathbf{a}_j p_j(t_i) \qquad (84)
\end{aligned}$$

and

$$\mathbf{P_a}(t_i) = E\{[\mathbf{a}-\hat{\mathbf{a}}(t_i)]\,[\mathbf{a}-\hat{\mathbf{a}}(t_i)]^{\mathrm{T}}\,|\mathbf{Z}(t_i) = \mathbf{Z}_i\} = \sum_{j=1}^{N_F} [\mathbf{a}_j-\hat{\mathbf{a}}(t_i)]\,[\mathbf{a}_j-\hat{\mathbf{a}}(t_i)]^{\mathrm{T}}\, p_j(t_i) \quad (85)$$

where $\mathbf{a}_j$ and p_j are the predefined parameter vector discrete value and its associated probability for a given elemental filter, for $j = 1, 2, ..., N_F$. It is important to note that neither of these calculations require knowledge of $\hat{\mathbf{x}}(t_i^+)$ or the associated state error covariance $\mathbf{P}(t_i^+)$ produced by the MMAE, as given by [44]:

$$\hat{\mathbf{x}}(t_i^+) = E\{\mathbf{x}(t_i)|\mathbf{Z}(t_i) = \mathbf{Z}_i\} = \sum_{j=1}^{N_F} \hat{\mathbf{x}}_j(t_i^+) p_j(t_i) \tag{86}$$

$$\begin{aligned} \mathbf{P}(t_i^+) &= E\left\{ \left[\mathbf{x}(t_i) - \hat{\mathbf{x}}(t_i^+)\right] \left[\mathbf{x}(t_i) - \hat{\mathbf{x}}(t_i^+)\right]^{\mathrm{T}} | \mathbf{Z}(t_i) = \mathbf{Z}_i \right\} \\ &= \sum_{j=1}^{N_F} p_j(t_i) \left\{ \mathbf{P}_j(t_i^+) + \left[\hat{\mathbf{x}}_j(t_i) - \hat{\mathbf{x}}(t_i^+)\right] \left[\hat{\mathbf{x}}_j(t_i) - \hat{\mathbf{x}}(t_i^+)\right]^{\mathrm{T}} \right\} \end{aligned} \tag{87}$$

where $\hat{\mathbf{x}}_j(t_i^+)$ is the Kalman filter-computed state estimate based on $\mathbf{a}_j$, and $\mathbf{P}_j(t_i^+)$ is the Kalman filter-computed state error covariance based on $\mathbf{a}_j$. Note that Equations (84) - (87) cannot be pre-computed since knowledge of the measurement history is required.

The goal of the M^{3}AE architecture is to provide accurate state and parameter estimation simultaneously. Therefore, given the MMAE-supplied parameter estimate $\hat{\mathbf{a}}$, and the measurements $\mathbf{z}$, a state estimator is required to generate $\hat{\mathbf{x}}_{M^3AE}$. Thus the M^{3}AE architecture, developed in this research, implements a standard Kalman Filter tuned for state estimation, based upon an assumed parameter value given by $\hat{\mathbf{a}}_{MMAE}(t_i)$.

Under the assumptions stated, the conditional state error covariance of a standard Kalman filter is equal to the unconditional covariance of the estimation error [43]. Thus, under normal circumstances, the covariance is completely precomputable, without any knowledge of the actual measurements taken. However, in this new architecture, an analysis is required to determine the impact of providing the state estimator an updated parameter estimate at each time sample. Therefore, an approximate evaluation of the error covariance committed by the new architecture is presented next,

with the intent to be able to predict the state estimation performance of the M^{3}AE before actually building the M^{3}AE and conducting a full-scale Monte Carlo analysis of it.

3.3.1 Evaluation of M^{3}AE Associated Errors

The objective of this section is to develop a good approximate evaluation of the state estimation error conditional covariance matrix for the new M^{3}AE architecture. The conditional covariance should provide a direct indication of how well the M^{3}AE architecture performs when affected by uncertain parameters. As stated above, this development assumes a Kalman filter form of the state estimator. It further assumes that the parameter values, $\mathbf{a}$, affecting the system are actually characterized by the true parameter values, $\mathbf{a}_\mathrm{T}$. Also note that the following development assumes parameter variations occur only in the $\mathbf{\Phi}$, $\mathbf{B}_\mathrm{d}$, or $\mathbf{H}$ system matrices. However, the formulation is readily extended to parameter variations in the $\mathbf{Q}_\mathrm{d}$ or $\mathbf{R}$ noise matrices, and this is presented later in Section 3.3.2.

The true system is described by

$$\mathbf{x}(t_i;\mathbf{a}_\mathrm{T}) = \mathbf{\Phi}(t_i,t_{i-1};\mathbf{a}_\mathrm{T})\mathbf{x}(t_{i-1};\mathbf{a}_\mathrm{T}) + \mathbf{B}_\mathrm{d}(t_{i-1};\mathbf{a}_\mathrm{T})\mathbf{u}(t_{i-1}) + \mathbf{G}_\mathrm{d}(t_{i-1})\mathbf{w}_\mathrm{d}(t_{i-1}) \quad (88)$$

$$\mathbf{z}(t_i;\mathbf{a}_\mathrm{T}) = \mathbf{H}(t_i;\mathbf{a}_\mathrm{T})\mathbf{x}(t_i;\mathbf{a}_\mathrm{T}) + \mathbf{v}(t_i) \quad (89)$$

and the Kalman filter based on parameter value $\mathbf{a}$ is described by

$$\widehat{\mathbf{x}}(t_i^-;\mathbf{a}) = \mathbf{\Phi}(t_i,t_{i-1};\mathbf{a})\widehat{\mathbf{x}}(t_{i-1}^+;\mathbf{a}) + \mathbf{B}_\mathrm{d}(t_{i-1};\mathbf{a})\mathbf{u}(t_{i-1}) \quad (90)$$

$$\widehat{\mathbf{x}}(t_i^+;\mathbf{a}) = \widehat{\mathbf{x}}(t_i^-;\mathbf{a}) + \mathbf{K}(t_i;\mathbf{a})\left[\mathbf{z}(t_i;\mathbf{a}_\mathrm{T}) - \mathbf{H}(t_i;\mathbf{a})\widehat{\mathbf{x}}(t_i^-;\mathbf{a})\right] \quad (91)$$

where

$$\mathbf{K}(t_i;\mathbf{a}) = \mathbf{P}(t_i^-;\mathbf{a})\mathbf{H}^\mathrm{T}(t_i;\mathbf{a})\mathbf{A}^{-1}(t_i;\mathbf{a})$$

and

$$\mathbf{A}(t_i;\mathbf{a}) = \mathbf{H}(t_i;\mathbf{a})\mathbf{P}(t_i^-;\mathbf{a})\mathbf{H}^\mathrm{T}(t_i;\mathbf{a}) + \mathbf{R}(t_i) \quad (92)$$

and where the propagation relations for the covariance are described by:

$$\mathbf{P}(t_i^-;\mathbf{a}) = \mathbf{\Phi}(t_i,t_{i-1};\mathbf{a})\mathbf{P}(t_{i-1}^+;\mathbf{a})\mathbf{\Phi}^{\mathrm{T}}(t_i,t_{i-1};\mathbf{a}) + \mathbf{G}_{\mathrm{d}}(t_{i-1})\mathbf{Q}_{\mathrm{d}}(t_{i-1})\mathbf{G}_{\mathrm{d}}^{\mathrm{T}}(t_{i-1}) \tag{93}$$

and the corresponding measurement update is given by

$$\mathbf{P}(t_i^+;\mathbf{a}) = \mathbf{P}(t_i^-;\mathbf{a}) - \mathbf{K}(t_i;\mathbf{a})\mathbf{H}(t_i;\mathbf{a})\mathbf{P}(t_i^-;\mathbf{a}) \tag{94}$$

Also, note that this development assumes that the control input is exactly known to the filter. For eventual implementation, a U-D covariance factorization form or square root filter [43, 44, 59, 60] might actually be used, but this research will continue to develop filter algorithms in the forms of Equations (90) – (94).

Given the above foundation and assumptions, the state estimation error of the new filter conditioned on $\mathbf{a}$, is defined by:

$$\mathbf{e}_{M^3AE}(t_i) \triangleq \hat{\mathbf{x}}_{M^3AE}(t_i) - \mathbf{x}_{True}(t_i) \tag{95}$$

where it is again emphasized that full order models are assumed in this development. Note that a generalized approach using reduced order filter models is possible, and the technique is discussed in [43], in which the state estimation error would have the form:

$$\mathbf{e}_{M^3AE\_reduced}(t_i) = \mathbf{C}(t_i)\hat{\mathbf{x}}_{M^3AE}(t_i) - \mathbf{C}_{True}(t_i)\mathbf{x}_{True}(t_i) \tag{96}$$

where $\mathbf{C}(t_i)$ is an c-by-n linear transformation and $\mathbf{C}_{True}(t_i)$ is an c-by-n_{T} linear transformation, with c being the critical quantities related to the filter states and n is the dimension of the filter design model, and n_{T} is the dimension of the truth model. Equation (96) with $c = n$ and $\mathbf{C}(t_i) = \mathbf{I}$ is most directly related to Equation (95).

The error before and after measurement update at the t_ith measurement is defined as follows:

$$\mathbf{e}_{M^3AE}(t_i^-) = \hat{\mathbf{x}}_{M^3AE}(t_i^-) - \mathbf{x}_{True}(t_i) \tag{97}$$

$$\mathbf{e}_{M^3AE}(t_i^+) = \hat{\mathbf{x}}_{M^3AE}(t_i^+) - \mathbf{x}_{True}(t_i) \tag{98}$$

and the error covariances before and after incorporating the t_ith measurement are denoted as $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i^-;\mathbf{a}_{\mathrm{T}},\mathbf{a})$ and $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i^+;\mathbf{a}_{\mathrm{T}},\mathbf{a})$, respectively, where this notation highlights the depen-

dence of these values on the particular choices of $\mathbf{a}_{\mathrm{T}}$ (T denotes *True*) in the truth model and $\mathbf{a}$ in the single Kalman filter state estimator within the M^{3}AE. The eventual goal is to evaluate or approximate these matrices explicitly for $\mathbf{a} = \hat{\mathbf{a}}_{MMAE}(t_i)$. These are given by:

$$\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i^-; \mathbf{a}_{\mathrm{T}}, \mathbf{a}) \triangleq E\{\mathbf{e}_{M^3AE}(t_i^-)\mathbf{e}_{M^3AE}(t_i^-)^{\mathrm{T}}|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\} - E\{\mathbf{e}_{M^3AE}(t_i^-)|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\}E\{\mathbf{e}_{M^3AE}(t_i^-)^{\mathrm{T}}|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\} \quad (99)$$

$$\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i^+; \mathbf{a}_{\mathrm{T}}, \mathbf{a}) \triangleq E\{\mathbf{e}_{M^3AE}(t_i^+)\mathbf{e}_{M^3AE}(t_i^+)^{\mathrm{T}}|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\} - E\{\mathbf{e}_{M^3AE}(t_i^+)|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\}E\{\mathbf{e}_{M^3AE}(t_i^+)^{\mathrm{T}}|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\} \quad (100)$$

Notice that, appropriate to a performance analysis, these are unconditional covariances versus conditioned on a specific sample of the measurement history vector. The notation $E\{\cdot\,|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\}$ is meant to convey the fact that the truth model is based on $\mathbf{a}_{\mathrm{T}}$ whereas the single state estimator within the M^{3}AE is based on $\mathbf{a}$ (eventually to be evaluated at $\mathbf{a} = \hat{\mathbf{a}}_{MMAE}(t_i)$). The first term on the right hand side of the covariance equations shown above is the correlation, $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_{\mathrm{T}}, \mathbf{a})$, at either t_i^- or t_i^+. The next section will focus on deriving a first order approximation to $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_{\mathrm{T}}, \mathbf{a})$, in order to develop a viable design tool for predicting system performance. The second term,

$$E\{\mathbf{e}_{M^3AE}(t_i^-)\}E\{\mathbf{e}_{M^3AE}(t_i^-)^{\mathrm{T}}|\mathbf{a}_{\mathrm{T}}, \mathbf{a}\}$$

is the mean of the error times itself transposed. It is a "bias-like" term, and as in other covariance performance analyses, may well be ignored for expediency [44]. However, its derivation and impact are presented in Section 3.3.1.3.

3.3.1.1 The Correlation, $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_{\mathrm{T}}, \hat{\mathbf{a}})$

This section derives a first order approximation to the correlation, $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_{\mathrm{T}}, \hat{\mathbf{a}})$, based on a truncated series representation of errors, and using a single sample of the measurement-time history data to evaluate some of the component expressions. This approximation will in turn be used as the basis for the approximate covariance analysis tool developed for the M^{3}AE architecture. The

approximations are required in order to develop a viable covariance analysis capability for design expediency. The alternative to this approximation is to conduct a time-consuming multi-run Monte Carlo analysis to determine $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_T, \hat{\mathbf{a}})$ properly, but it is desired to have an approximate covariance of errors prior to such a full-scale Monte Carlo evaluation of the M^{3}AE, in order to enhance iterative design decision-making.

The parameter estimation error is defined as:

$$\mathbf{e}_{\mathbf{a}}(t_i) \triangleq \hat{\mathbf{a}}(t_i) - \mathbf{a}_T(t_i) \tag{101}$$

where, as stated earlier, it is assumed that the parameter values do not change from one time sample to the next. Given that the parameter estimate vector is supplied by an MMAE, the parameter estimation error term is the same term presented in [44], where the conditional covariance of the parameter was shown to be

$$\mathbf{P}_{\mathbf{a}}(t_i) = E\{[\mathbf{a}_T - \hat{\mathbf{a}}(t_i)]\,[\mathbf{a}_T - \hat{\mathbf{a}}(t_i)]^T \,|\mathbf{Z}(t_i) = \mathbf{Z}_i\} = \sum_{j=1}^{N_F} [\mathbf{a}_{Tj} - \hat{\mathbf{a}}(t_i)]\,[\mathbf{a}_{Tj} - \hat{\mathbf{a}}(t_i)]^T\, p_j(t_i) \tag{102}$$

where $p_j(t_i)$ is the hypothesis conditional probability associated with the j-th elemental filter within the MMAE:

$$\begin{aligned} p_j(t_i) &\triangleq \text{Prob}\,\{\mathbf{a} = \mathbf{a}_j | \mathbf{Z}(t_i) = \mathbf{Z}_i\} \\ &= \frac{f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\mathbf{a}_j, \mathbf{Z}_{i-1}) p_j(t_{i-1})}{\sum_{k=1}^{N_F} f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\mathbf{a}_k, \mathbf{Z}_{i-1}) p_k(t_{i-1})} \,. \end{aligned} \tag{103}$$

starting from some appropriate initial condition, such as $p_j(t_0) = \frac{1}{N_F}$ for $j = 1, 2, ..., N_F$.

Given this foundation for the new M^{3}AE architecture, a first order approximation to the state estimation error described by Equation (95) is developed. The goal of using this approximation is to develop a viable design tool to predict potential system performance before a full-scale Monte

Carlo analysis of the M^{3}AE is conducted. This first order error approximation is given by a truncated series representation, truncated to first order. Letting

$$\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_T} = [\hat{\mathbf{x}}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_T} - \mathbf{x}_T(t_i)] \tag{104}$$

be the state estimate error in a Kalman filter based on the true parameter value, the following expression is obtained:

$$\begin{aligned} \mathbf{e}_{M^3AE}(t_i) &\cong \mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_T} + \sum_{k=1}^{N_P} \frac{\partial \mathbf{e}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_T} [\widehat{a}_k(t_i) - a_{Tk}(t_i)] \\ &= \mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_T} + \sum_{k=1}^{N_P} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_T} e_{a_k}(t_i) \end{aligned} \tag{105}$$

since $\frac{\partial \mathbf{x}_T(t_i)}{\partial a_k} = 0$. N_P is the number of uncertain parameters involved in the problem (i.e., the dimensionality of the parameter space).

The first term on the right is the error in the state estimate coming out of a single Kalman filter assuming that the true values of the parameters are known exactly. Given the assumptions stated for this development, the mean of this error is zero [43]. The second term on the right accounts for the fact that the parameters are not exactly known. This first order error expression (Equation (105)) is used to develop the correlation approximation below.

The correlation approximation of the error is, to first order,

$$\Psi_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_T, \hat{\mathbf{a}}) \cong E\left\{ \begin{array}{l} \left(\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_T} + \sum_{k=1}^{N_P} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_T} e_{a_k}(t_i) \right) * \\ \left(\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_T} + \sum_{l=1}^{N_P} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_T} e_{a_l}(t_i) \right)^T \end{array} \right\} \tag{106}$$

Equation (106) can be rewritten as:

$$\begin{aligned}\boldsymbol{\Psi}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}) &\cong E\left\{(\mathbf{m}+\mathbf{n})(\mathbf{m}+\mathbf{n})^\mathrm{T}\right\} \\ &= E\left\{\mathbf{m}\mathbf{m}^\mathrm{T}+\mathbf{m}\mathbf{n}^\mathrm{T}+\mathbf{n}\mathbf{m}^\mathrm{T}+\mathbf{n}\mathbf{n}^\mathrm{T}\right\} \\ &= E\{\mathbf{m}\mathbf{m}^\mathrm{T}\}+E\{\mathbf{m}\mathbf{n}^\mathrm{T}\}+E\{\mathbf{n}\mathbf{m}^\mathrm{T}\}+E\{\mathbf{n}\mathbf{n}^\mathrm{T}\} \qquad (107)\end{aligned}$$

where $\mathbf{m}=\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}$ and $\mathbf{n}=\sum_{k=1}^{N_P}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}e_{a_k}(t_i)$.

The first term on the right in Equation (107) is

$$E\{\mathbf{m}\mathbf{m}^\mathrm{T}\}=E\left\{(\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}})(\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}})^\mathrm{T}\right\} \qquad (108)$$

which is equivalent to the conditional covariance of the state estimate error

$$\begin{aligned}\mathbf{P}_{\mathbf{mm}} &= E\left\{(\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}})(\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}})^\mathrm{T}\right\}- \\ &\quad E\left\{\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}E\left\{\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}^\mathrm{T}\right\} \qquad (109)\end{aligned}$$

(since $E\left\{\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}=\mathbf{0}$) when the parameter vector values are the true values. This leads to the standard Kalman Filter propagate and update equations given below:

$$\begin{aligned}\mathbf{P}_{\mathbf{mm}}(t_i^-;\mathbf{a}_\mathrm{T}) &= \boldsymbol{\Phi}(t_i,t_{i-1};\mathbf{a}_\mathrm{T})\mathbf{P}(t_{i-1}^+;\mathbf{a}_\mathrm{T})\boldsymbol{\Phi}^\mathrm{T}(t_i,t_{i-1};\mathbf{a}_\mathrm{T}) \\ &\quad +\mathbf{G}_\mathrm{d}(t_{i-1})\mathbf{Q}_\mathrm{d}(t_{i-1})\mathbf{G}_\mathrm{d}^\mathrm{T}(t_{i-1}) \qquad (110)\end{aligned}$$

$$\mathbf{P}_{\mathbf{mm}}(t_i^+;\mathbf{a}_\mathrm{T})=\mathbf{P}(t_i^-;\mathbf{a}_\mathrm{T})-\mathbf{K}(t_i;\mathbf{a}_\mathrm{T})\mathbf{H}(t_i;\mathbf{a}_\mathrm{T})\mathbf{P}(t_i^-;\mathbf{a}_\mathrm{T}) \qquad (111)$$

If the cross correlation terms of Equation (107) are zero, then the overall correlation, $\boldsymbol{\Psi}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_\mathrm{T},\hat{\mathbf{a}})$, will be simplified greatly. Therefore, consider the third term on the right side of Equation (107):

$$E\{\mathbf{n}\mathbf{m}^\mathrm{T}\} = E\left\{\left(\sum_{k=1}^{N_P}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}e_{a_k}(t_i)\right)(\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}})^\mathrm{T}\right\} \qquad (112)$$

In considering Equation (112), note that $e_{a_k}(t_i)$ as defined in Equation (101) is a direct function of the measurement history random vector, $\mathbf{Z}(t_i)$, whereas $[\mathrm{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}]$ is independent of $\mathbf{Z}(t_i)$ [43]. Thus, given that these latter two random variables are independent, the following Theorem from [52] applies:

"Let Y_1 and Y_2 be independent random variables with joint density $f(y_1, y_2)$. Let $g(Y_1)$ and $h(Y_2)$ be functions of Y_1 and Y_2, respectively. Then

$$E[g(Y_1)\,h(Y_2)] = E[g(Y_1)]\,E[h(Y_2)] \tag{113}$$

provided the expectations exist."

Therefore, since $\mathrm{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}$ is independent of $\mathbf{Z}(t_i)$, it is also uncorrelated with a function of $\mathbf{Z}(t_i)$, namely $e_{a_k}(t_i)$. Thus

$$\begin{aligned} E\{\mathbf{n}\mathbf{m}^\mathrm{T}\} &= E\left\{\left(\sum_{k=1}^{N_P}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}} e_{a_k}(t_i)\right)\left(\mathrm{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right)^\mathrm{T}\right\} \\ &= E\left\{\sum_{k=1}^{N_P}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}} e_{a_k}(t_i)\right\} E\left\{\mathrm{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}^\mathrm{T}\right\} \\ &= \mathbf{0} \end{aligned} \tag{114}$$

since

$$E\{\mathrm{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\} = \mathbf{0}$$

Consequently, the cross correlation terms ($E\{\mathbf{m}\mathbf{n}^\mathrm{T}\}$ and $E\{\mathbf{n}\mathbf{m}^\mathrm{T}\}$) of Equation (107) are zero, simplifying $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_\mathrm{T}, \hat{\mathbf{a}})$.

Finally, the last term on the right hand side of Equation (107) leads to:

$$E\{\mathbf{n}\mathbf{n}^{\mathrm{T}}\} = E\left\{ \begin{array}{l} \left(\sum_{k=1}^{N_P} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}} e_{a_k}(t_i)\right) * \\ \left(\sum_{l=1}^{N_P} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial al}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}} e_{a_l}(t_i)\right)^{\mathrm{T}} \end{array} \right\} \quad (115)$$

$$= E\left(\sum_{k=1}^{N_P}\sum_{l=1}^{N_P} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)^{\mathrm{T}}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}} e_{a_k}(t_i) e_{a_l}(t_i)\right)$$

$$= \sum_{k=1}^{N_P}\sum_{l=1}^{N_P} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)^{\mathrm{T}}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}} e_{a_k}(t_i) e_{a_l}(t_i)\right\}$$

$$= \sum_{k=1}^{N_P}\sum_{l=1}^{N_P} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k} \frac{\partial \hat{\mathbf{x}}_{KF}(t_i)^{\mathrm{T}}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\} E\left\{e_{a_k}(t_i) e_{a_l}(t_i)\right\} \quad (116)$$

using the same type of uncorrelatedness arguments stated earlier, where $E\left\{e_{a_k}(t_i)e_{a_l}(t_i)\right\}$ is equivalent to the k,l^{th} (for all $k = 1,..., N_P$, and $l = 1,..., N_P$) entry of the correlation matrix, $E\{\mathbf{e_a}(t_i)\mathbf{e_a}(t_i)^{\mathrm{T}}\}$. This matrix is now approximated with the conditional expectation of the same quantities, conditioned on a single sample of the measurement time history:

$$E\{\mathbf{e_a}(t_i)\mathbf{e_a}(t_i)^{\mathrm{T}}\} \cong E\left\{\left[\mathbf{a}_{\mathrm{T}}-\hat{\mathbf{a}}(t_i)\right]\left[\mathbf{a}_{\mathrm{T}}-\hat{\mathbf{a}}(t_i)\right]^{\mathrm{T}}|\mathbf{Z}(t_i,\omega_r)=\mathbf{Z}_i\right\}$$

$$= \sum_{j=1}^{N_F}\left[\mathbf{a}_{\mathrm{T}j}-\hat{\mathbf{a}}(t_i)\right]\left[\mathbf{a}_{\mathrm{T}j}-\hat{\mathbf{a}}(t_i)\right]^{\mathrm{T}} p_j(t_i) \quad (117)$$

where the last equality is from Equation (102), and where $\hat{\mathbf{a}}(t_i)$ is given by Equation (84) and $p_j(t_i)$ comes from Equation (103). Also, in Equation (116), the $E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)^{\mathrm{T}}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\}$ term has been previously developed in [42,44] and is presented in Section 3.3.1.2.

Note that the above development focused on the first term (the correlation) of the covariance

$$\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_{\mathrm{T}},\hat{\mathbf{a}}) = \mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_{\mathrm{T}},\hat{\mathbf{a}}) -$$

$$E\{\mathbf{e}_{M^3AE}(t_i^-)|\mathbf{a}_{\mathrm{T}},\hat{\mathbf{a}}\}E\{\mathbf{e}_{M^3AE}(t_i^-)^{\mathrm{T}}|\mathbf{a}_{\mathrm{T}},\hat{\mathbf{a}}\} \quad (118)$$

where there is no conditioning on a single sample of the *entire* measurement history $\mathbf{Z}(t_i, \omega_r) = \mathbf{Z}_i$ (where ω_r is a single elemental outcome of the experiment) within the resulting expressions, the unconditional covariance, $\mathbf{P_a}(t_i)$, was approximated with $\mathbf{P_{a|Z}}(t_i|\mathbf{Z}(t_i) = \mathbf{Z}_i)$ (where the ω_r term has been suppressed from this point forward) as computed on only a *single sample* of measurement-time history data from a single Monte Carlo simulation of the MMAE. Alternatively, the $\mathbf{P_{a|Z}}(t_i|\mathbf{Z}(t_i) = \mathbf{Z}_i)$ values could be averaged over a multi-run Monte Carlo simulation to form a sample-statistic approximation to $\mathbf{P_a}(t_i)$ that is a better approximation to the true ensemble average than $\mathbf{P_{a|Z}}(t_i|\mathbf{Z}(t_i) = \mathbf{Z}_i)$ itself (i.e., averaging over the N_R runs, or conceptually running an approximation to the integration as $\mathbf{P_a}(t) = \int \left[\mathbf{P_{a|Z}}(t|\mathbf{Z}(t) = \mathbf{Z})\right] f_{\mathbf{Z}}(\mathbf{Z}(t))d\mathbf{Z}$), but this is undesirable because of tool expediency. It is desired to be able to evaluate Equation (118) approximately *without* requiring a full-scale Monte Carlo simulation. Otherwise, the designer may as well do a complete multi-run Monte Carlo analysis of the M^{3}AE! Therefore, the M^{3}AE approximate covariance analysis tool approximates $\mathbf{P_a}(t_i)$ with the $\mathbf{P_{a|Z}}(t_i|\mathbf{Z}(t_i) = \mathbf{Z}_i)$, which is the *best* representation of $\mathbf{P_a}(t_i)$ for all t_i that can be computed by the MMAE on a one-run Monte Carlo simulation.

Also note that there is no corresponding *filter-computed* indication of the mean error from a single Monte Carlo run, in distinction to the filter-computed $\mathbf{P_{a|Z}}(t_i|\mathbf{Z}(t_i) = \mathbf{Z}_i)$. Therefore an approximate mean error value is not available, so $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a_T}, \hat{\mathbf{a}})$ may serve as an approximation as well as an upper bound to $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a_T}, \hat{\mathbf{a}})$, and in turn provides a good approximation to the expected performance for a proposed M^{3}AE. However, as stated earlier, Equation (116) is not precomputable since $\hat{\mathbf{a}}(t_i)$ and $p_j(t_i)$ are required at each sample time. Thus a single Monte Carlo run of the MMAE-based parameter estimator is required to produce $\hat{\mathbf{a}}(t_i)$, $\mathbf{P_a}(t_i)$, and $p_j(t_i)$ for input into this algorithm. Once accomplished, the M^{3}AE approximate covariance analysis may be conducted. So in summary, the final equation used in the M^{3}AE approximate covariance analysis is

$$\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}) \cong \mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_\mathrm{T},\hat{\mathbf{a}})$$

$$= RHS \text{ of Eq. (108)} + RHS \text{ of Eq. (116)} \qquad (119)$$

The recursions associated with these terms are presented in the next section. Section 3.3.1.3 then presents an alternative method for evaluating the last term in Equation (118), rather than invoking the approximation of Equation (119). The M^{3}AE covariance analysis tool itself is described in more detail in Section 3.4

3.3.1.2 Approximate Covariance Recursions: Uncertainties in Φ, $\mathbf{B}_\mathrm{d}$, and $\mathbf{H}$

This section presents the complete propagation and update recursions pertaining to $E\left\{\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)^\mathrm{T}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}$ for use in Equation (116). The recursions were originally developed to provide the iterative maximum likelihood solution to the simultaneous state and parameter estimation problem, accounting for possible parameter variations in the Φ, $\mathbf{B}_\mathrm{d}$, and, $\mathbf{H}$ system matrices [42]. These equations are not intended for on-line usage of the M^{3}AE, but to provide an approximate indication of potential performance before the M^{3}AE is actually implemented. This development parallels that of [42, 44].

The recursions are developed in the following manner. First the standard Kalman filter equations are listed, followed by their partials with respect to the uncertain parameter, a_k, for $k = 1, 2, ..., N_P$. Once such partials are evaluated, products of such partials can be formed and expectations can be taken, to generate $E\left\{\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)^\mathrm{T}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}$. Finally, the expressions needed for the covariance matrix recursions are presented.

The standard Kalman filter propagation equations are:

$$\widehat{\mathbf{x}}_{KF}(t_i^-) = \mathbf{\Phi}(t_i, t_{i-1})\widehat{\mathbf{x}}_{KF}(t_{i-1}^+) + \mathbf{B}_{\mathrm{d}}(t_{i-1})\mathbf{u}(t_{i-1}) \tag{120}$$

$$\mathbf{P}(t_i^-) = \mathbf{\Phi}(t_i, t_{i-1})\mathbf{P}(t_{i-1}^+)\mathbf{\Phi}^{\mathrm{T}}(t_i, t_{i-1}) + \mathbf{G}_{\mathrm{d}}(t_{i-1})\mathbf{Q}_{\mathrm{d}}(t_{i-1})\mathbf{G}_{\mathrm{d}}^{\mathrm{T}}(t_{i-1}) \tag{121}$$

$$\mathbf{A}(t_i) = \mathbf{H}(t_i)\mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i) + \mathbf{R}(t_i) \tag{122}$$

where Equation (122) is included here for convenience, rather than in the update equations. Their associated partials (suppressing the time arguments equal to t_{i-1} and the KF subscript on $\widehat{\mathbf{x}}$ on the right hand sides of the following equations) are readily formed as:

$$\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)}{\partial a_k} = \mathbf{\Phi}\frac{\partial \hat{\mathbf{x}}^+}{\partial a_k} + \frac{\partial \mathbf{\Phi}}{\partial a_k}\hat{\mathbf{x}}^+ + \frac{\partial \mathbf{B}_{\mathrm{d}}}{\partial a_k}\mathbf{u} \tag{123}$$

$$\frac{\partial \mathbf{P}(t_i^-)}{\partial a_k} = \frac{\partial \mathbf{\Phi}}{\partial a_k}\mathbf{P}^+\mathbf{\Phi}^{\mathrm{T}} + \mathbf{\Phi}\frac{\partial \mathbf{P}^+}{\partial a_k}\mathbf{\Phi}^{\mathrm{T}} + \mathbf{\Phi}\mathbf{P}^+\frac{\partial \mathbf{\Phi}^{\mathrm{T}}}{\partial a_k} \tag{124}$$

$$\frac{\partial \mathbf{A}(t_i)}{\partial a_k} = \mathbf{H}(t_i)\frac{\partial \mathbf{P}(t_i^-)}{\partial a_k}\mathbf{H}^{\mathrm{T}}(t_i) + \frac{\partial \mathbf{H}(t_i)}{\partial a_k}\mathbf{P}(t_i^-)\mathbf{H}(t_i)^{\mathrm{T}} + \mathbf{H}(t_i)\mathbf{P}(t_i^-)\frac{\partial \mathbf{H}^{\mathrm{T}}(t_i)}{\partial a_k} \tag{125}$$

for $k = 1, 2, ..., N_P$. Note that $\frac{\partial \mathbf{\Phi}}{\partial a_k}$ and $\frac{\partial \mathbf{B}_{\mathrm{d}}}{\partial a_k}$ must be known for all k and $\frac{\partial \mathbf{u}}{\partial a_k} = 0$, since at time t_i, control inputs through time t_{i-1} have already been applied and cannot be changed, even for feedback control [44].

Finally, the desired relations for propagating the expressions for the covariance matrix forward in time are (suppressing the time arguments equal to t_{i-1}, the KF subscript from $\widehat{\mathbf{x}}_{KF}$, and shortening "$\mathbf{a} = \mathbf{a}_{\mathrm{T}}$" to "$\mathbf{a}_{\mathrm{T}}$" on the right hand sides of the following equations):

$$E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)}{\partial a_k}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)^{\mathrm{T}}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\}$$

$$\begin{aligned}
= \; & \boldsymbol{\Phi} E\left\{\frac{\partial \hat{\mathbf{x}}^+}{\partial a_k}\frac{\partial \hat{\mathbf{x}}^{+\mathrm{T}}}{\partial a_l}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}} + \frac{\partial \boldsymbol{\Phi}}{\partial a_k} E\left\{\hat{\mathbf{x}}^+\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \boldsymbol{\Phi}^{\mathrm{T}}}{\partial a_l} \\
& + \boldsymbol{\Phi} E\left\{\frac{\partial \hat{\mathbf{x}}^+}{\partial a_k}\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \boldsymbol{\Phi}^{\mathrm{T}}}{\partial a_l} + \frac{\partial \boldsymbol{\Phi}}{\partial a_k} E\left\{\hat{\mathbf{x}}^+\frac{\partial \hat{\mathbf{x}}^{+\mathrm{T}}}{\partial a_l}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}} \\
& + \frac{\partial \mathbf{B}_{\mathrm{d}}}{\partial a_k} E\left\{\mathbf{u}\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \mathbf{B}_{\mathrm{d}}^{\mathrm{T}}}{\partial a_l} \\
& + \frac{\partial \boldsymbol{\Phi}}{\partial a_k} E\left\{\hat{\mathbf{x}}^+\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \mathbf{B}_{\mathrm{d}}^{\mathrm{T}}}{\partial a_l} + \frac{\partial \mathbf{B}_{\mathrm{d}}}{\partial a_k} E\left\{\mathbf{u}\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \boldsymbol{\Phi}^{\mathrm{T}}}{\partial a_l} \\
& + \boldsymbol{\Phi} E\left\{\frac{\partial \hat{\mathbf{x}}^+}{\partial a_k}\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \mathbf{B}_{\mathrm{d}}^{\mathrm{T}}}{\partial a_l} + \frac{\partial \mathbf{B}_{\mathrm{d}}}{\partial a_k} E\left\{\mathbf{u}\frac{\partial \hat{\mathbf{x}}^{+\mathrm{T}}}{\partial a_l}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}}
\end{aligned} \tag{126}$$

$$\begin{aligned}
E\left\{\hat{\mathbf{x}}(t_i^-)\hat{\mathbf{x}}(t_i^-)^{\mathrm{T}}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\} = \; & \boldsymbol{\Phi} E\left\{\hat{\mathbf{x}}^+\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}} + \mathbf{B}_{\mathrm{d}} E\left\{\mathbf{u}\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{B}_{\mathrm{d}}^{\mathrm{T}} \\
& + \boldsymbol{\Phi} E\left\{\hat{\mathbf{x}}^+\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{B}_{\mathrm{d}}^{\mathrm{T}} + \mathbf{B}_{\mathrm{d}} E\left\{\mathbf{u}\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}}
\end{aligned} \tag{127}$$

$$\begin{aligned}
E\left\{\frac{\partial \hat{\mathbf{x}}(t_i^-)}{\partial a_k}\hat{\mathbf{x}}(t_i^-)^{\mathrm{T}}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\} = \; & \boldsymbol{\Phi} E\left\{\frac{\partial \hat{\mathbf{x}}^+}{\partial a_k}\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}} + \frac{\partial \boldsymbol{\Phi}}{\partial a_k} E\left\{\hat{\mathbf{x}}^+\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}} \\
& + \boldsymbol{\Phi} E\left\{\frac{\partial \hat{\mathbf{x}}^+}{\partial a_k}\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{B}_{\mathrm{d}}^{\mathrm{T}} + \frac{\partial \mathbf{B}_{\mathrm{d}}}{\partial a_k} E\left\{\mathbf{u}\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{B}_{\mathrm{d}}^{\mathrm{T}} \\
& + \frac{\partial \boldsymbol{\Phi}}{\partial a_k} E\left\{\hat{\mathbf{x}}^+\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{B}_{\mathrm{d}}^{\mathrm{T}} \\
& + \frac{\partial \mathbf{B}_{\mathrm{d}}}{\partial a_k} E\left\{\mathbf{u}\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\boldsymbol{\Phi}^{\mathrm{T}}
\end{aligned} \tag{128}$$

where Equations (127) and (128) are required to define certain terms in Equation (126). Also note that, once Equation (128) is computed, its transpose will also be used. If the expectations involving $\mathbf{u}(t_i)$ are required, they are evaluated in one of two ways [44]. If $\mathbf{u}(t_i)$ is completely precomputed

(especially if it is zero), then

$$E\left\{\mathbf{u}\ (\ )^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\} \Longrightarrow \mathbf{u}E\left\{(\ )^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\} \tag{129}$$

and the following recursions are required to evaluate all necessary terms (suppressing the time arguments equal to t_{i-1}, the KF subscript from $\hat{\mathbf{x}}_{KF}$, and shortening "$\mathbf{a} = \mathbf{a}_{\mathrm{T}}$" to "$\mathbf{a}_{\mathrm{T}}$" on the right hand sides of the following equations):

$$E\left\{\hat{\mathbf{x}}_{KF}(t_i^+)|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\} = \mathbf{\Phi}E\left\{\hat{\mathbf{x}}^+|_{\mathbf{a}_{\mathrm{T}}}\right\} + \mathbf{B}_{\mathrm{d}}\mathbf{u} \tag{130}$$

$$\begin{aligned} E\left\{\frac{\partial\hat{\mathbf{x}}_{KF}(t_i^+)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\} &= \mathbf{D}(t_i)\left[\mathbf{\Phi}E\left\{\frac{\partial\hat{\mathbf{x}}^+}{\partial a_k}|_{\mathbf{a}_{\mathrm{T}}}\right\} + \frac{\partial\mathbf{\Phi}}{\partial a_k}E\left\{\hat{\mathbf{x}}^+|_{\mathbf{a}_{\mathrm{T}}}\right\} + \frac{\partial\mathbf{B}_{\mathrm{d}}}{\partial a_k}\mathbf{u}\right] \\ &- \mathbf{K}(t_i)\frac{\partial\mathbf{H}(t_i)}{\partial a_k}E\left\{\hat{\mathbf{x}}(t_i^+)|_{\mathbf{a}_{\mathrm{T}}}\right\} \end{aligned} \tag{131}$$

where $\mathbf{D}(t_i) = \mathbf{I} - \mathbf{K}(t_i)\mathbf{H}(t_i)$. But if feedback control is implemented, then $\mathbf{u}(t_i) = -\mathbf{C}(t_i)\hat{\mathbf{x}}(t_i^+)$, and the expectations become:

$$E\left\{\mathbf{u}(\ )^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\} \Longrightarrow -\mathbf{C}E\left\{\hat{\mathbf{x}}^+(\ )^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\} \tag{132}$$

$$E\left\{\mathbf{u}\mathbf{u}^{\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\} \Longrightarrow \mathbf{C}E\left\{\hat{\mathbf{x}}^+\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{C}^{\mathrm{T}} \tag{133}$$

for which recursions have already been developed.

To incorporate the measurement at time t_i, the standard Kalman filter state equations are:

$$\mathbf{K}(t_i) = \mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i)\mathbf{A}^{-1}(t_i) \tag{134}$$

$$\mathbf{r}_j = \left[\mathbf{z}_i - \mathbf{H}(t_i)\hat{\mathbf{x}}_{KF}(t_i^-)\right] \tag{135}$$

$$\hat{\mathbf{x}}_{KF}(t_i^+) = \hat{\mathbf{x}}_{KF}(t_i^-) + \mathbf{K}(t_i)\mathbf{r}_j \tag{136}$$

$$\mathbf{P}(t_i^+) = \mathbf{P}(t_i^-) - \mathbf{K}(t_i)\mathbf{H}(t_i)\mathbf{P}(t_i^-) \tag{137}$$

$$= \mathbf{D}(t_i)\mathbf{P}(t_i^-)\mathbf{D}^{\mathrm{T}}(t_i) + \mathbf{K}(t_i)\mathbf{R}(t_i)\mathbf{K}^{\mathrm{T}}(t_i) \tag{138}$$

and their associated partials are (suppressing the KF subscript on $\hat{\mathbf{x}}_{KF}$ on the right hand sides of the following equations):

$$\begin{aligned}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^+)}{\partial a_k} &= \mathbf{D}(t_i)\left\{\frac{\partial \hat{\mathbf{x}}(t_i^-)}{\partial a_k} + \left(\frac{\partial \mathbf{P}(t_i^-)}{\partial a_k}\mathbf{H}^{\mathrm{T}}(t_i) + \mathbf{P}(t_i^-)\frac{\partial \mathbf{H}^{\mathrm{T}}(t_i)}{\partial a_k}\right)\mathbf{A}^{-1}(t_i)\mathbf{r}_j\right\} \\ &- \mathbf{K}(t_i)\frac{\partial \mathbf{H}(t_i)}{\partial a_k}\hat{\mathbf{x}}(t_i^+) \end{aligned} \tag{139}$$

$$\frac{\partial \mathbf{P}(t_i^+)}{\partial a_k} = \mathbf{D}(t_i)\frac{\partial \mathbf{P}(t_i^-)}{\partial a_k}\mathbf{D}^{\mathrm{T}}(t_i) - \mathbf{P}(t_i^+)\frac{\partial \mathbf{H}^{\mathrm{T}}(t_i)}{\partial a_k}\mathbf{K}^{\mathrm{T}}(t_i) - \mathbf{K}(t_i)\frac{\partial \mathbf{H}(t_i)}{\partial a_k}\mathbf{P}(t_i^+) \tag{140}$$

Again, taking products of terms as in Equation (139) and applying the expectation operation yields the desired covariance matrix relations as (suppressing the time arguments equal to t_i, the KF subscript on $\hat{\mathbf{x}}_{KF}$, and shortening "$\mathbf{a} = \mathbf{a}_{\mathrm{T}}$" to "$\mathbf{a}_{\mathrm{T}}$" on the right hand sides of the following equations) [44]:

$$\begin{aligned} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^+)}{\partial a_k}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^+)^{\mathrm{T}}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\} &= \mathbf{D}E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)}{\partial a_k}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)^{\mathrm{T}}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{D}^{\mathrm{T}} \\ &+ \left[\mathbf{D}\left(\frac{\partial \mathbf{P}^-}{\partial a_k}\mathbf{H}^{\mathrm{T}} + \mathbf{P}^-\frac{\partial \mathbf{H}^{\mathrm{T}}}{\partial a_k}\right) - \mathbf{K}\frac{\partial \mathbf{H}}{\partial a_k}\mathbf{P}^-\mathbf{H}^{\mathrm{T}}\right]\mathbf{A}^{-1} \\ &* \left[\mathbf{D}\left(\frac{\partial \mathbf{P}^-}{\partial a_l}\mathbf{H}^{\mathrm{T}} + \mathbf{P}^-\frac{\partial \mathbf{H}^{\mathrm{T}}}{\partial a_l}\right) - \mathbf{K}\frac{\partial \mathbf{H}}{\partial a_l}\mathbf{P}^-\mathbf{H}^{\mathrm{T}}\right]^{\mathrm{T}} \\ &+ \mathbf{D}E\left\{\frac{\partial \hat{\mathbf{x}}^-}{\partial a_k}\hat{\mathbf{x}}^{-\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \mathbf{H}^{\mathrm{T}}}{\partial a_l}\mathbf{K}^{\mathrm{T}} \\ &+ \mathbf{K}\frac{\partial \mathbf{H}}{\partial a_k}E\left\{\hat{\mathbf{x}}^-\frac{\partial \hat{\mathbf{x}}^{-\mathrm{T}}}{\partial a_l}|_{\mathbf{a}_{\mathrm{T}}}\right\}\mathbf{D}^{\mathrm{T}} \\ &+ \mathbf{K}\frac{\partial \mathbf{H}}{\partial a_k}E\left\{\hat{\mathbf{x}}^-\hat{\mathbf{x}}^{-\mathrm{T}}|_{\mathbf{a}_{\mathrm{T}}}\right\}\frac{\partial \mathbf{H}^{\mathrm{T}}}{\partial a_l}\mathbf{K}^{\mathrm{T}} \end{aligned} \tag{141}$$

$$E\left\{\frac{\partial\hat{\mathbf{x}}^{+}}{\partial a_k}\hat{\mathbf{x}}^{+\mathrm{T}}|_{\mathbf{a}_\mathrm{T}}\right\} = \mathbf{D}\left[E\left\{\frac{\partial\hat{\mathbf{x}}^{-}}{\partial a_k}\hat{\mathbf{x}}^{-\mathrm{T}}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}+\left(\frac{\partial\mathbf{P}^{-}}{\partial a_k}\mathbf{H}^{\mathrm{T}}+\mathbf{P}^{-}\frac{\partial\mathbf{H}^{\mathrm{T}}}{\partial a_k}\right)\mathbf{K}^{\mathrm{T}}\right]- \mathbf{K}\frac{\partial\mathbf{H}}{\partial a_k}\left(E\left\{\hat{\mathbf{x}}^{-}\hat{\mathbf{x}}^{-\mathrm{T}}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}+\mathbf{P}^{-}\mathbf{H}^{\mathrm{T}}\mathbf{K}^{\mathrm{T}}\right) \tag{142}$$

$$E\left\{\hat{\mathbf{x}}(t_i^{+})\hat{\mathbf{x}}(t_i^{+})^{\mathrm{T}}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\} = E\left\{\hat{\mathbf{x}}(t_i^{-})\hat{\mathbf{x}}(t_i^{-})^{\mathrm{T}}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}+\mathbf{K}\mathbf{A}\mathbf{K}^{\mathrm{T}} \tag{143}$$

3.3.1.3 *Bias Term,* $E\{\mathbf{e}_{M^3AE}(t_i)|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\}E\{\mathbf{e}_{M^3AE}(t_i)^{\mathrm{T}}|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\}$

The first term of the conditional covariance, $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i)$, of Equations (99) and (100) was determined in Section 3.3.1. All that remains is evaluation of the bias term $E\{\mathbf{e}_{M^3AE}(t_i)|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\}*$ $E\{\mathbf{e}_{M^3AE}(t_i)^{\mathrm{T}}|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\}$. This term shall be developed here, although it may not be used in the M^{3}AE approximate covariance analysis if one chooses to lower the computational burden by using Equation (119) in place of Equation (118). Looking at the first component of this term yields the following, using Equation (105):

$$\begin{aligned}E\{\mathbf{e}_{M^3AE}(t_i)|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\} &\cong E\left\{\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}+\sum_{k=1}^{N_P}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}e_{a_k}(t_i)\right\}\\ &= E\left\{\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}+E\left\{\sum_{k=1}^{N_P}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}e_{a_k}(t_i)\right\}\\ &= E\left\{\sum_{k=1}^{N_P}\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}e_{a_k}(t_i)\right\}\end{aligned} \tag{144}$$

since $E\left\{\mathbf{e}_{KF}(t_i)|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}=\mathbf{0}$. Therefore

$$\begin{aligned}E\{\mathbf{e}_{M^3AE}(t_i)|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\} &\cong \sum_{k=1}^{N_P}E\left\{\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}e_{a_k}(t_i)\right\}\\ &= \sum_{k=1}^{N_P}E\left\{\frac{\partial\hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}E\left\{e_{a_k}(t_i)\right\}\end{aligned} \tag{145}$$

Note the first term on the right-hand-side of Equation (145) was previously presented in Equation (131). If either term $E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_T}\right\}$ or $E\{e_{a_k}(t_i)\}$ were zero, the task would be simple and complete, since then $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_T,\hat{\mathbf{a}}) = \mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_T,\hat{\mathbf{a}})$, as already computed. In other words, the prior development in Section 3.3 would provide not just an upper bound on $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i)$, ignoring the nonzero second term in Equations (99) and (100), but would be a more accurate representation since the second terms would truly be zero. However, in general, neither $\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}|_{\mathbf{a}=\mathbf{a}_T}$ nor $e_{a_k}(t_i)$ are zero mean processes. Nevertheless, two special cases involving the $e_{a_k}(t_i)$ term warrant further discussion.

Case 1. If the parameter space is only made up of discrete points, such that the true parameter may only assume one of the discrete points at any time, and the MMAE is set up such that it has that point in its filter bank, then $e_{a_k}(t_i)$ will have a zero mean error *in steady state* [4]. This is due to the fact that the MMAE $\hat{\mathbf{a}}$ will converge to that true parameter value under these conditions [4]. Note, however, this situation does not occur for most problems of interest.

Case 2. Given the more practical case, where a discretized parameter space is considered, Baram [4] showed that, under the assumptions stated in Section 3.2, the MMAE output will converge to the filter which is "closest" to the true value from a specific information distance measure standpoint. The error, $e_{a_k}(t_i)$, may not be zero, but its *steady state value* can be found directly by simply subtracting the "closest" filter's assumed value of $\mathbf{a}_j = \mathbf{a}_C$ (C for "closest") from the true value of $\mathbf{a}_T(t_i)$ and then taking the k-th scalar component. Thus

$$\begin{aligned} E\{e_{a_k}(t_i)\} &= E\{a_{Ck}(t_i) - a_{Tk}(t_i)\} \\ &= a_{Ck}(t_i) - a_{Tk}(t_i) \end{aligned} \tag{146}$$

since $a_{Ck}(t_i)$ and $a_{Tk}(t_i)$ are known. Therefore, given the discussion above, Equation (146) is equivalent to:

$$e_{a_k}(t_i) = a_{Ck}(t_i) - a_{Tk}(t_i) \tag{147}$$

Therefore, the "known" value of $e_{a_k}(t_i)$ replaces $E\{e_{a_k}(t_i)\}$and is used in Equation (145) directly. This argument also holds for MMAE's which place lower bounds on the elemental filter probabilities to avoid estimator lockup. For this type of MMAE, the $\hat{\mathbf{a}}(t_i)$ computed by Equation (59) is subtracted from the true value of $\mathbf{a}_T(t_i)$ to determine $e_{a_k}(t_i)$ for $k = 1, 2, ..., N_P$.

If it is desired to account for the "bias" term, $E\{\mathbf{e}_{M^3AE}(t_i)|\mathbf{a}_T, \hat{\mathbf{a}}\}E\{\mathbf{e}_{M^3AE}(t_i)^T|\mathbf{a}_T, \hat{\mathbf{a}}\}$, Equation (145) and its transpose are used to write (suppressing the conditioning, "$|\mathbf{a}_T, \hat{\mathbf{a}}$" from both sides of the equation for convenience):

$$\begin{aligned} E\{\mathbf{e}_{M^3AE}(t_i)\}E\{\mathbf{e}_{M^3AE}(t_i)^T\} &= \sum_{k=1}^{N_P}\sum_{l=1}^{N_P} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}\right\} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)^T}{\partial a_l}\right\} * \\ & \quad E\{e_{a_k}(t_i)\}\, E\{e_{a_l}(t_i)\} \qquad (148) \\ &= \sum_{k=1}^{N_P}\sum_{l=1}^{N_P} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)}{\partial a_k}\right\} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i)^T}{\partial a_l}\right\} (e_{a_k}(t_i)e_{a_l}(t_i)) \qquad (149) \end{aligned}$$

As mentioned earlier, mean effects are generally present, but often ignored in covariance performance analyses, since it is mainly desired to get an upper bound on potential performance in as quick a manner as possible. Therefore, the M^{3}AE approximate covariance analysis tool can ignore $E\{\mathbf{e}_{M^3AE}(t_i)\}E\{\mathbf{e}_{M^3AE}(t_i)^T\}$ and use just $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_T, \hat{\mathbf{a}})$ versus $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i; \mathbf{a}_T, \hat{\mathbf{a}})$, i.e., using Equation (119) versus Equation (118).

3.3.2 Uncertainties in $\mathbf{Q}_d$ and $\mathbf{R}$

The development of Section 3.3.1.2 is now modified to address design problems in which the uncertain parameters affect either the dynamic noise covariance, $\mathbf{Q}_d$, or the measurement noise

covariance, $\mathbf{R}$. The only difference in the algorithms presented here versus Section 3.3.1.2 lies in the evaluation of the partial derivatives in each relationship. The equations in this section are much simpler to develop and implement. Again this presentation will parallel [44]. The computations for propagating forward in time are simply (again, suppressing the time arguments equal to t_{i-1} and shortening "$\mathbf{a} = \mathbf{a}_\mathrm{T}$" to "$\mathbf{a}_\mathrm{T}$" on the right-hand-side of equations):

$$E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)}{\partial a_k}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)^\mathrm{T}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\} = \mathbf{\Phi} E\left\{\frac{\partial \hat{\mathbf{x}}^+}{\partial a_k}\frac{\partial \hat{\mathbf{x}}^{+\mathrm{T}}}{\partial a_l}|_{\mathbf{a}_\mathrm{T}}\right\}\mathbf{\Phi}^\mathrm{T} \tag{150}$$

which is only the first term from Equation (126). Equations (127) and (128) are no longer required, and

$$\frac{\partial \mathbf{P}(t_i^-)}{\partial a_k} = \mathbf{\Phi}\frac{\partial \mathbf{P}^+}{\partial a_k}\mathbf{\Phi}^\mathrm{T} + \mathbf{G}_\mathrm{d}\frac{\partial \mathbf{Q}_\mathrm{d}}{\partial a_k}\mathbf{G}_\mathrm{d}^\mathrm{T} \tag{151}$$

Incorporating the measurement update, yields the following relations [44]:

$$\begin{aligned} E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^+)}{\partial a_k}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^+)^\mathrm{T}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\} &= \mathbf{D}E\left\{\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)}{\partial a_k}\frac{\partial \hat{\mathbf{x}}_{KF}(t_i^-)^\mathrm{T}}{\partial a_l}|_{\mathbf{a}=\mathbf{a}_\mathrm{T}}\right\}\mathbf{D}^\mathrm{T} \\ &\quad + \frac{\partial \mathbf{E}(t_i)}{\partial a_k}\mathbf{A}^{-1}(t_i)\frac{\partial \mathbf{E}(t_i)}{\partial a_l} \end{aligned} \tag{152}$$

to replace Equation (141), where

$$\frac{\partial \mathbf{E}(t_i)}{\partial a_k} = \frac{\partial \mathbf{P}(t_i^-)}{\partial a_k}\mathbf{H}^\mathrm{T}(t_i) - \mathbf{K}(t_i)\frac{\partial \mathbf{A}(t_i)}{\partial a_k} \tag{153}$$

$$\frac{\partial \mathbf{A}(t_i)}{\partial a_k} = \mathbf{H}(t_i)\frac{\partial \mathbf{P}(t_i^-)}{\partial a_k}\mathbf{H}^\mathrm{T}(t_i) + \frac{\partial \mathbf{R}(t_i)}{\partial a_k} \tag{154}$$

$$\frac{\partial \mathbf{P}(t_i^+)}{\partial a_k} = \mathbf{D}(t_i)\frac{\partial \mathbf{P}(t_i^-)}{\partial a_k}\mathbf{D}^\mathrm{T}(t_i) + \mathbf{K}(t_i)\frac{\partial \mathbf{R}(t_i)}{\partial a_k}\mathbf{K}^\mathrm{T}(t_i) \tag{155}$$

and Equations (142) and (143) are no longer required.

3.4 M^{3}AE Design Tool

This section discusses in more detail how the designer would use the approximate covariance analysis tool developed for the M^{3}AE architecture. This tool gives the designer the ability to analyze, tune, and predict system performance before the M^{3}AE is constructed and subjected to a full-scale Monte Carlo analysis. Figure 12 is a repeat of Figure 3 and shows in general how a designer developing a system based on the M^{3}AE architecture could conduct an approximate covariance analysis after only a single Monte Carlo run on just the MMAE-based parameter estimator within the M^{3}AE.

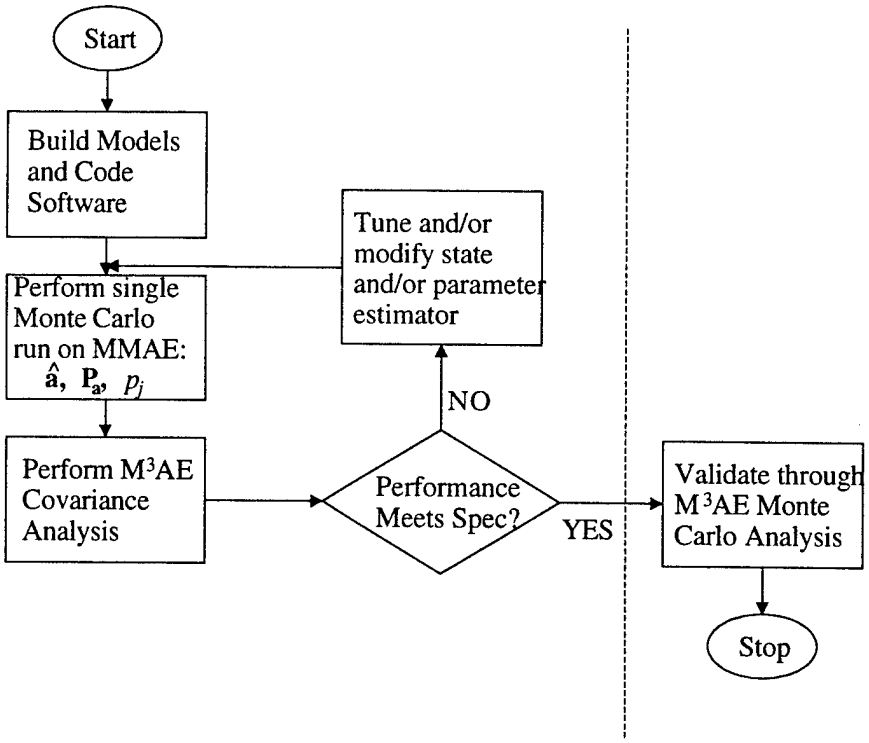


Figure 12. Performance Analysis Tool for M^{3}AE

The designer should accomplish the following steps for a viable design:

1. Design the MMAE based on Sheldon's 5-step algorithm for parameter estimation presented in Section 2.2. Note that the constant-gain, steady state filter restriction has been eliminated by using a finite horizon assumption and conducting a constrained optimization, as described more fully in Section 3.5. The finite horizon limits the number of the iterations required to generate the solution to the discrete time Lyapunov equation presented in Section 2.2. The length of the horizon is *ad hoc* and based on what finite period of time (finite horizon) is of physical concern for the problem. Section 3.5 discusses these modifications to Sheldon's work.

2. Enhance the fixed-bank MMAE design by using Lund's IRDF techniques applied to discrete-time problems (see Section 2.3). The algorithm for implementing IRDF for discrete-time systems is presented in Section 3.6.

3. Design the Kalman filter for state estimation using the common techniques discussed in [43, 44]. However, in designing this Kalman filter, it is important to realize that the Kalman filter will receive real-time, parameter estimates from the MMAE, in addition to the measurements, at each sample time.

4. Code the MMAE in software and conduct a single Monte Carlo run to generate the time histories of the parameter estimates, $\hat{\mathbf{a}}(t_i)$, filter-computed $\mathbf{P_a}(t_i)$, and the associated elemental filter probabilities, $p_j(t_i)$.

5. Code the state estimator system matrices and associated system noises into the M^{3}AE covariance analysis tool.

6. Perform the covariance analysis.

7. If the performance meet specifications, conduct a thorough multi-run Monte Carlo analysis on the entire M^{3}AE design. If performance does not meet specification, tune and/or modify the state and/or parameter estimator (using a greater number N_F of points in the parameter space discretization, for example), depending on where the problem areas exist. For example, if an event detection design was generating too many false alarms, then investigations into the MMAE would be appropriate. Moreover, if state estimation precision was not meeting specifications, the structure of the second term on the right-hand-side of Equation (119) (namely, Equation (116)) provides useful insights into which parameters should receive the designer's attention in order to provide the desired benefit to particular states of concern. Such *sensitivity* information is especially useful for making good design decisions.

After modification and/or tuning is accomplished, Steps 4-7 should be repeated until performance meets specifications.

3.5 Enhanced Sheldon Parameter Optimization

As discussed in Section 2.2, Sheldon's original development assumes steady state, constant-gain filters (i.e., $\mathbf{K}_j$ does not vary with time). Sheldon focused on performance at steady state, since, at steady state, the MMAE *would have* converged to the "best" discrete parameter value (in the Baram distance measure sense). If problems with time invariant system models and stationary noises are considered only, then the steady state Kalman filter will be a constant-gain Kalman filter. However, the filters in the MMAE do not have to be steady state Kalman filters, but for the development to apply, the filters must have the same structure as a Kalman filter. Therefore, any other method to calculate a constant gain that produces a stable estimator could be used [70]. If the linear (or linearized) system model for a given application is astable or unstable, then steady state conditions are not achievable. Instead, a physically motivated finite horizon is used for assessing

performance, and it is assumed that the MMAE will converge to the "best" parameter estimate (best in the Baram distance measure sense) within that finite horizon period of time. The length of the finite horizon is motivated by the designer's physical insights into how many sample periods into the future the discretization algorithm should look when performing its vector minimization. Thus, for the second problem researched in Chapter 4 (an unstable, nonlinear, integrated GPS/INS problem using extended Kalman filters), a nominal trajectory point is chosen as the basis for defining the system matrices for determining a "*pseudo*"-constant gain value. Once the gains are calculated, the Sheldon 5-step algorithm may be applied using a finite horizon assumption to generate an approximate solution, since the system never achieves steady state. Thus the designer must decide the length of the finite horizon for the problem at hand. The finite horizon assumption is implemented in software during the generation of the solution to the discrete time Lyapunov equation

$$E\left\{\begin{bmatrix}\tilde{\mathbf{x}}_j^- \\ \mathbf{x}_T^-\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{x}}_j^{-\mathrm{T}} & \mathbf{x}_T^{-\mathrm{T}}\end{bmatrix}\right\} = \mathbf{\Gamma}_j(t_i) = \mathbf{\Upsilon}\mathbf{\Gamma}_j(t_{i-1})\mathbf{\Upsilon}^{\mathrm{T}} + \mathbf{G}_0\mathbf{Q}_0\mathbf{G}_0^{\mathrm{T}} \qquad (156)$$

presented earlier in Section 2.2. Thus, the finite horizon limits the number of the iterations required to determine an adequate solution.

Finally, Sheldon's cost functional for parameter estimation, given in Equation (46), is minimized using a constrained-range optimization technique. The constrained-range optimization technique restricts the search range of the minimization to the admissible parameter space. This allows the minimization to search in a smaller, acceptable region (which is problem dependent), for the "best" values to minimize the cost functional.

This enhanced Sheldon algorithm was co-developed with Vasquez [77] and implemented using MATLAB [40] for the examples given in Chapter 4. MATLAB code was developed and written

to generate the required parameter sets for each example. A modified Simpson's rule [77] was the numerical integration technique used, and the vector minimization was accomplished using MATLAB's constrained optimization techniques which are based on Sequential Quadratic Programming (SQP). Specifically, the command "*constr*" in MATLAB's Optimization Toolbox performs constrained nonlinear optimization for multivariable functions [39].

3.6 Discrete-Time IRDF

This section develops the Lund algorithm for the sampled-data measurement case. Lund's original development for the continuous-time measurement case was presented in Section 2.3. Specifically, this section develops the algorithm for implementing the discrete-time modulation parameter, $\eta(t_i)$, that is analogous to $\eta(t)$ of the original Lund IRDF concept.

Recall from Section 2.3, that a general way to keep the inter-residual distance measure, $J_{jk}(t_i)$, above some specified limit was to vary the dynamics noise covariances, $\mathbf{Q}_{\mathrm{d}j}(t_i)$, and thus adjust the filter gains. For the sampled-data measurement case, the filter propagation and update equations for covariances and gains are

$$\mathbf{P}_j(t_{i+1}^-) = \mathbf{\Phi}_j \mathbf{P}_j(t_i^+) \mathbf{\Phi}_j^{\mathrm{T}} + \mathbf{G}_{\mathrm{d}j}(t_i) \mathbf{Q}_{\mathrm{d}j}(t_i) \mathbf{G}_{\mathrm{d}j}^{\mathrm{T}}(t_i) \tag{157}$$

$$\mathbf{P}_j(t_{i+1}^+) = \mathbf{P}_j(t_{i+1}^-) - \mathbf{K}_j(t_{i+1}) \mathbf{H}_j(t_{i+1}) \mathbf{P}_j(t_{i+1}^-) \tag{158}$$

$$\mathbf{K}_j(t_{i+1}) = \mathbf{P}_j(t_{i+1}^-) \mathbf{H}_j^{\mathrm{T}}(t_{i+1}) \left[\mathbf{H}_j(t_{i+1}) \mathbf{P}_j(t_{i+1}^-) \mathbf{H}_j^{\mathrm{T}}(t_{i+1}) + \mathbf{R}_j(t_{i+1}) \right]^{-1} \tag{159}$$

and $\mathbf{Q}_{\mathrm{d}j}(t_i)$ is replaced by

$$\mathbf{Q}'_{\mathrm{d}j}(t_i) = \eta(t_i) \mathbf{Q}_{\mathrm{d}j}(t_i), \; j = 1, \ldots, N_F \tag{160}$$

where the modulating parameter, $\eta(t_i) \in [\eta_{\min}, 1.0]$, which is analogous to the continuous-time measurement case (see Section 2.3 for discussion on $\eta(t)$).

Lund's proposed simplification of modulating the new information $\mathbf{K}_j(t_i)\mathbf{r}_j(t_i)$ by $\eta(t_i)$ for the sampled-data measurement case is shown by the following equation:

$$\mathbf{K}'_j(t_i)\mathbf{r}_j(t_i) = \eta(t_i)\mathbf{K}_j(t_i)\mathbf{r}_j(t_i) \tag{161}$$

The filter gains are now precomputable and only the modulation is computed on-line [35].

Additionally, in Lund's continuous-time measurement development, the time derivative of the modulating variable $\eta(t)$ was given in Section 2.3 by Equation (67). For the discrete-time-measurement equations, the derivative of $\eta(t)$ is replaced by the increment of $\eta(t_i)$ defined as

$$\begin{aligned} \triangle\eta(t_i) &= \xi\left\{J_{jk}(t_i) - J^0_{jk}(t_i)\right\}, \quad \text{Cond 1} \\ &= 0, \quad \text{Cond 2} \end{aligned} \tag{162}$$

where

$$\begin{aligned} &\text{Cond 1}: \quad \eta(t_i) \in [\eta_{\min}, 1.0] \\ &\text{Cond 2}: \quad \eta(t_i) = \eta_{\min} \text{ AND } \xi\left\{J_{jk}(t_i) - J^0_{jk}(t_i)\right\} < 0 \quad \text{OR} \\ &\qquad\quad \eta(t_i) = 1 \text{ AND } \xi\left\{J_{jk}(t_i) - J^0_{jk}(t_i)\right\} > 0 \end{aligned} \tag{163}$$

As in the continuous-time measurement case, "Condition 2" operations again provide anti-windup compensation. Note that the $\eta_{\min}$, ξ, and J^0_{jk} values are determined in the same manner as for the continuous-time measurement case presented in Section 2.3.

At each update cycle, the new modulation value is defined as:

$$\eta(t_i) = \eta(t_{i-1}) + \triangle\eta(t_i) \tag{164}$$

The residuals are also required at t_i to compute $\triangle\eta(t_i)$. Then this computed $\eta(t_i)$ gets applied to Equation (160) and incorporated into Equation (157) for the next propagation to t_{i+1}. The algorithm for implementing this discrete-time modulation value is shown in Figure 13. The algorithm first picks the two elemental filters which are the "closest" to the true value. This is determined by looking at the elemental filter probabilities, $p_j(t_i)$. The two filters with the greatest $p_j(t_i)$'s are considered the closest in the "Baram sense" [4] to the true value. IRDF's goal is to ensure that these two elemental filters are as "distant" as possible from each other (increasing distinguishability

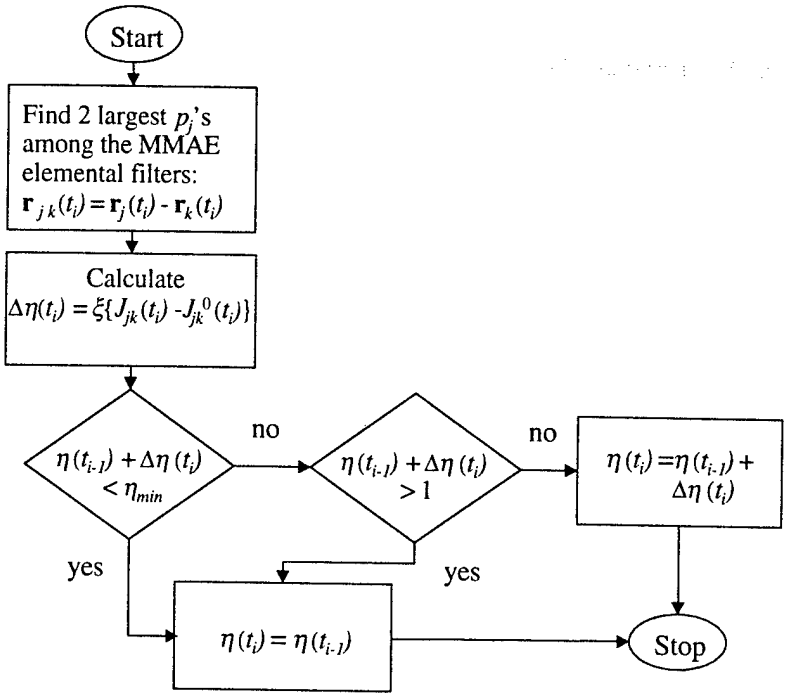


Figure 13. Modulation Parameter Calculation

between filters) by modulating filter gains. Equation (162) is used to calculate $\triangle\eta(t_i)$ at each time sample. Once $\triangle\eta(t_i)$ is determined, it is compared against the conditions listed in Equation (163). If Condition 1 is met, $\triangle\eta(t_i)$ is added to the previous value of $\eta(t_{i-1})$ according to Equation (164). If Condition 2 is met, the value of η doesn't change:

$$\eta(t_i) = \eta(t_{i-1}) \tag{165}$$

Then $\eta(t_i)$ is used to modulate either the dynamic driving noise covariance $\mathbf{Q}_{\mathrm{d}j}(t_i)$ or the new information $\mathbf{K}_j(t_i)\mathbf{r}_j(t_i)$ as discussed above. Therefore, Equation (157) becomes:

$$\mathbf{P}_j(t_{i+1}^-) = \mathbf{\Phi}_j\mathbf{P}_j(t_i^+)\mathbf{\Phi}_j^{\mathrm{T}} + \mathbf{G}_{\mathrm{d}j}(t_i)\mathbf{Q}'_{\mathrm{d}j}(t_i)\mathbf{G}_{\mathrm{d}j}^{\mathrm{T}}(t_i) \tag{166}$$

or $\mathbf{K}'_j(t_i)$ replaces $\mathbf{K}_j(t_i)$ in the measurement update equations. The discrete time difference equations are used in the research examples presented in Chapter 4.

3.7 Summary

The M^{3}AE architecture has been developed in this Chapter. It is based on a fixed-bank MMAE used for parameter estimation and a single Kalman filter used for state estimation. The placement of the MMAE's discrete parameter points (used as the basis for defining elemental filters) is chosen by using Sheldon's parameter optimization algorithm [69], modified herein to be applicable to a wider class of problems than originally proposed. The distinguishability of the elemental filters has been further enhanced by employing Lund's IRDF techniques applied to the discrete-time, fixed-bank MMAE [35].

Additionally, a valuable approximate covariance analysis design tool has been developed to assist the designer in predicting potential performance for an M^{3}AE-based system. Parameter variations in any of the $\mathbf{\Phi}$, $\mathbf{B}_{\mathrm{d}}$, $\mathbf{G}_{\mathrm{d}}$, $\mathbf{H}$, $\mathbf{Q}_{\mathrm{d}}$, and $\mathbf{R}$ system matrices are possible. An approximate covariance analysis may be accomplished after conducting only a single Monte Carlo run on the MMAE-based parameter estimator. The "square of the mean" term subtracted from an error corre-

lation matrix to form the error covariance, $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i)$, can be ignored for most problems, especially if one is interested in expediency and only an upper bound on performance.

Two examples are presented in Chapter 4, to illustrate the effectiveness of the covariance analysis tool, as well as highlight the overall performance of an M^3AE-based system.

Chapter 4 - M^{3}AE Evaluation

The new M^{3}AE architecture solves the problem of estimating states and parameters simultaneously. To demonstrate the M^{3}AE's performance improvement over previously published results involving conventional MMAE's, two examples are developed and their results presented in this chapter [69, 78]. The first example involves a simple second-order mechanical translational system, in which the system's undamped natural frequency is the uncertain parameter [69]. The second example involves a 13-state nonlinear integrated GPS/INS system, in which measurement noise affecting the GPS outputs is the uncertain parameter [78, 79]. The results demonstrate application of the theory and performance predictions discussed in previous chapters.

This chapter is separated into two major sections – one for each example. Both examples use MMAE banks made up of three elemental filters, where the state models used in the elemental filters are the same as the state model used in the single Kalman filter (used for state estimation) within the M^{3}AE algorithm. Additionally, there is no order reduction between the truth and filter models, and only one parameter value is changing in each example. Both examples are implemented in FORTRAN using the Multiple Model Simulation for Optimal Filter Evaluation (MMSOFE) software package [58–60,62,63,78]. Sheldon's optimum parameter discretization algorithm, the M^{3}AE approximate covariance analysis tool, and data reduction is accomplished using MATLAB [39,40].

4.1 Second Order System with Uncertainties in Φ and $\mathbf{Q}_d$

The second order problem presented below is the same example Sheldon considered in his research [69, 70]. The uncertain parameter is present in $\mathbf{F}$, $\mathbf{B}$, and $\mathbf{G}$ of the continuous-time model, and thus in the Φ, $\mathbf{B}_d$ and $\mathbf{Q}_d$ matrices of the equivalent discrete-time model [43], as will be shown. However in this example, the input $\mathbf{u}(t)$ is assumed $\mathbf{0}$. Therefore, the impact of an uncertain parameter in $\mathbf{B}_d$ is not investigated in this example. This section is presented as follows:

1. System description – since each example comes from past research, the problem descriptions will be taken from previously published works.

2. M^{3}AE design tool implementation, results, and analysis – discussion of the results from the 7-step algorithm of Section 3.4. Additionally, the impacts of implementing the enhanced Sheldon's optimization algorithms and Lund's discrete-time IRDF in the M^{3}AE architecture are analyzed. Finally, the section finishes with a comparison between the M^{3}AE approximate covariance analysis and actual results.

3. M^{3}AE Analysis – includes comparisons between actual results from a 10-run Monte Carlo simulation of the M^{3}AE and that of a conventional fixed-bank MMAE ("Sheldon-discretized" for state estimation, as opposed to the MMAE within the M^{3}AE that is "Sheldon-discretized" for parameter estimation).

4. Summary – summarizes the major themes discovered during the analysis of the test case.

4.1.1 System Description

The following system description comes directly from [69]. The true system is an ideal mechanical translational system. It is a continuous-time system which is modelled by the second order linear equation:

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) \tag{167}$$

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{k}{m} \end{bmatrix} u(t) \tag{168}$$

where x_1 is the position in meters and x_2 is the velocity in meters per second. Restating in more general terms, ω_n denotes the undamped natural frequency, $\sqrt{\frac{k}{m}}$, and ζ denotes the damping ratio, $\frac{b}{2\sqrt{km}}$.

To frame the system in a stochastic setting, a dynamics driving noise term is added to the model to produce

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) + \mathbf{G}(t)\mathbf{w}(t) \tag{169}$$

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix} w(t) \tag{170}$$

with

$$E\{w(t)w(t+\tau)\} = Q\delta(\tau)$$

and

$$Q = 0.01 \text{ cm}^2 \text{ sec}$$

The units of Q may seem strange at first glance since w has units of cm/sec^2, but they are valid since ω_n^2 has units of radians^2/sec^2 and $\delta(\tau)$ carries units of (time)$^{-1}$. The damping ratio is set to $\zeta = 0.01$ and the natural frequency, ω_n, is allowed to vary from 2π (~ 6.28) to 20π (~ 62.83) radians per second. These values of Q, ζ, and ω_n, were chosen to accentuate the differences between models by increasing the sensitivity to changes in natural frequency. Larger damping ratios such as $\zeta = 0.707$ result in a system for which there is a less distinguishable difference between models based on different ω_n values, so a multiple model technique may not be warranted. The Q is chosen to provide enough system energy for convergence of the adaptation process, yet not so much as to swamp out the system with noise effects.

The system is implemented with an equivalent discrete-time model of the form [43]:

$$\mathbf{x}(t_i) = \mathbf{\Phi}(t_i, t_{i-1})\mathbf{x}(t_{i-1}) + \mathbf{B}_\mathrm{d}(t_{i-1})\mathbf{u}(t_{i-1}) + \mathbf{G}_\mathrm{d}(t_{i-1})\mathbf{w}_\mathrm{d}(t_{i-1}) \tag{171}$$

where

$$\begin{array}{lcl} t_i - t_{i-1} & = & 0.01 \text{ sec } \forall\, i \\ \mathbf{\Phi}(t_i, t_{i-1}) & = & \text{the state transition matrix associated with } \mathbf{F}(t) \\ \mathbf{B}_\mathrm{d}(t_{i-1}) & = & \int_{t_{i-1}}^{t_i} \mathbf{\Phi}(t_i, \tau)\mathbf{B}(\tau)\mathrm{d}\tau \end{array}$$

and $\mathbf{w}_\mathrm{d}(t_{i-1})$ is a discrete-time white Gaussian noise process with zero mean and covariance kernel

$$E\left\{[\mathbf{w}_{\mathrm{d}}(t_{i-1})]\,[\mathbf{w}_{\mathrm{d}}(t_j)]^{\mathrm{T}}\right\} = \begin{cases} \mathbf{Q}_{\mathrm{d}}(t_{i-1}) & t_{i-1} = t_j \\ \mathbf{0} & t_{i-1} \neq t_j \end{cases}$$

where

$$\mathbf{Q}_{\mathrm{d}}(t_{i-1}) = \int_{t_{i-1}}^{t_i} \mathbf{\Phi}(t_i,\tau)\mathbf{G}(\tau)Q(\tau)\mathbf{G}(\tau)^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}}(t_i,\tau)d\tau \tag{172}$$

Noisy measurements are available at each sample instant and are described by the equation

$$\mathbf{z}(t_i) = \mathbf{H}(t_i)\mathbf{x}(t_i) + v(t_i) \tag{173}$$

$$\mathbf{z}(t_i) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t_i) \\ x_2(t_i) \end{bmatrix} + v(t_i) \tag{174}$$

where $v(t_i)$ is a zero-mean white Gaussian noise process with covariance kernel

$$E\left\{[v(t_i)]\,[v(t_j)]\right\} = \begin{cases} R(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j \end{cases}$$

and

$$R(t_i) = 0.01 \text{ cm}^2 \qquad \forall\, i$$

Notice the truth model has the same structure as the filter models, with the only difference being in the value of ω_n, therefore the transformation matrix $\mathbf{T}$ of Equation (52) is the identity matrix. Additionally, a lower bound probability of $p_{\min} = 0.01$ is used in all the test cases shown. The value of 0.01 was determined experimentally. In general, too small a lower probability bound may slow the response of the MMAE to the onset of a changing parameter, while larger lower bound values may reduce the total amount of probability available for assignment to the MMAE's elemental filter with the "closest" $\mathbf{a}_j$ to the true parameter $\mathbf{a}_{\mathrm{T}}$, in the "Baram distance measure sense."

Finally, the state model used in the M^{3}AE's single Kalman filter (which has a purpose of accurate state estimation) is the same as the MMAE's elemental filters. The only difference is that

the single Kalman filter code has been modified to accept the M^{3}AE's MMAE-supplied parameter estimate at each sample time.

4.1.2 M^{3}AE Design Tool Implementation, Results, and Analysis

This sections describes the results from implementing the M^{3}AE 7-step design process of Section 3.4. Additionally, the impacts of implementing the enhanced Sheldon optimization algorithm and Lund's discrete-time IRDF on the M^{3}AE are investigated and analyzed. Finally, a comparison between the M^{3}AE approximate covariance analysis and actual results from a 10-run Monte Carlo simulation is presented.

4.1.2.1 Sheldon Parameter Optimization Algorithm

Sheldon's parameter optimization algorithm (see Section 2.2 and 3.5 for algorithm details) is implemented in MATLAB [40]. For this problem, both an unconstrained and constrained-range optimization technique are employed. After a comparison of the two methodologies, the constrained-range optimization (using "*constr*" from MATLAB's Optimization Toolbox [39]) using a 50-sample-period finite horizon is implemented in the optimization to determine the elemental filter parameter values used in this research example, even though the constrained-range optimization and finite horizon are not required for this second order, linear, stable system (Table 1 compares the results of both optimization techniques). The motivation to do so is twofold. The first reason is expediency, since the constrained-range optimization and finite horizon limited the parameter search space and the number of iterations required to solve the Lyapunov equation shown in Equation (53) during the minimization. The second reason is that it provides verification of the above implementation, since the second example to follow involves a 13-state nonlinear, unstable system in which the constrained-range optimization and finite horizon are required to determine the appropriate elemental filter parameter values. Additionally, as mentioned above, for the test cases examined in this

example, a lower bound ($p_{\min} = 0.01$) is placed on the elemental filter probabilities. Therefore, Equation (59) is used to solve for the $\hat{\mathbf{a}}(t_i)$ used in the cost function in Equation (46).

The constrained-range optimization with the 50-sample finite horizon produces results which are comparable to the unconstrained optimization, as shown in Table 1 below.

Table 1. Parameter Choices for State and Parameter Estimation

State Estimation Case		
	ω_n (rad/sec)	
Filter	Unconstrained	Constrained
1	18.51	22.88
2	35.30	37.89
3	53.09	54.43
Parameter Estimation Case		
	ω_n (rad/sec)	
Filter	Unconstrained	Constrained
1	12.77	13.44
2	28.64	28.40
3	49.67	49.90

The results from the constrained-range optimization are used in this example. Figure 14 displays the results of the constrained-range optimization used in the parameter estimation case. The valleys in the curve provide the optimal locations for placing the elemental filter $\mathbf{a}_j$'s. However, if $\mathbf{a}_{\mathrm{T}}$ is *near* any of the peaks in the curve, it is anticipated that parameter and state estimation performance will be poor [69]. Furthermore, notice the differences in the parameter discretization values between the optimized state estimation versus the optimized parameter estimation cases. As Sheldon points out in his research [69], the potential exists for the true parameter, $\mathbf{a}_{\mathrm{T}}$, to be "closer" in the "Baram distance measure sense" to an elemental filter's $\mathbf{a}_j$ in the MMAE "Sheldon-discretized" for optimized state estimation performance than to an elemental filter's $\mathbf{a}_j$ in an MMAE "Sheldon-discretized" for optimized parameter estimation performance (note that from this point forward, the following three

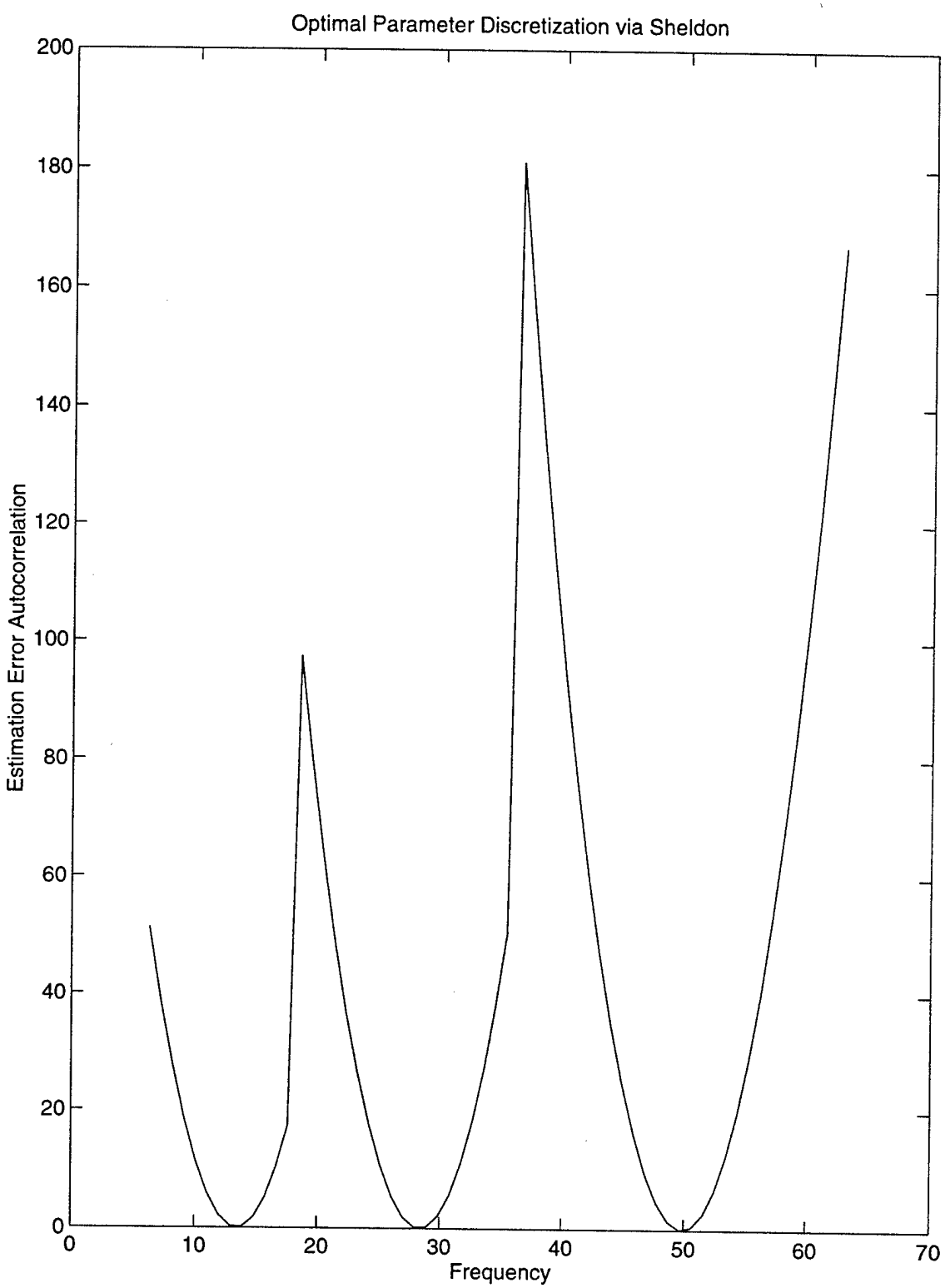


Figure 14. Sheldon's Autocorrelation Curve For Constrained-Range Parameter Discretization

abbreviations are used for convenience: *closest* – "closest" in the "Baram distance measure sense"; *MMAE/SDSEP* – MMAE "Sheldon-discretized" for optimized state estimation performance; and *MMAE/SDPEP* – MMAE "Sheldon-discretized" for optimized parameter estimation performance). If this were to occur, the *MMAE/SDSEP* will produce more accurate parameter estimates as well as more accurate state estimates than an *MMAE/SDPEP*. Therefore, it is expected for these cases, that the M^{3}AE may not perform as well as a conventional *MMAE/SDSEP*. Such a case is investigated and the results are discussed in Section 4.1.3.

4.1.2.2 Lund's Discrete-Time IRDF

The M^{3}AE's MMAE is implemented in the Multiple Model Simulation for Optimal Filter Evaluation (MMSOFE) [62] software program. MMSOFE is a FORTRAN-based program developed for evaluating various types of MMAEs.

An initial run of the "Sheldon-discretized" MMAE developed above is conducted to determine an approximate value for the inter-residual differences between the MMAE elemental filters, and thus to determine the inter-residual distance measure J_{jk}^0. The minimum modulation parameter, $\eta_{\min}$, and the attenuation, ξ, are chosen following the guideline presented in Section 2.3. If the system remains stable, [35] recommends setting $\eta_{\min}$ equal to zero, otherwise, iterating on $\eta_{\min}$ downward from one by orders of magnitude is recommended to get a "good" $\eta_{\min}$. Next the initial J_{jk}^0 chosen is the average value of the inter-residual differences between filters once good performance (in distinguishability) is achieved. Finally, ξ is chosen to provide proper attenuation of the $\eta(t_i)$. In this case, ξ [on the order of 0.7] provides a useful smoothing to Equation (162), whereas significantly smaller values cause undue sluggishness of response. Then a sensitivity analysis is conducted using various combinations of J_{jk}^0, $\eta_{\min}$, and ξ until the desired performance is achieved. For this problem, it was observed that larger values of ξ ($0.7 < \xi < 1$) and J_{jk}^0 ($3 < J_{jk}^0 < 5$) lead

to degraded M^{3}AE's state estimation performance without any improvement in parameter estimation performance, while smaller values of ξ ($\xi < 0.7$) and J_{jk}^0 ($J_{jk}^0 < 3$) had negligible impact on the state and parameter estimation performance. (In fact it was difficult to distinguish the performance between an M^{3}AE's with IRDF versus an M^{3}AE's without IRDF in both state and parameter estimation performance.) Table 2 displays the final results for this example.

Table 2. Lund IRDF Tuning Parameters

	J_{jk}^0	$\eta_{\min}$	ξ
Values Chosen	3.0	0.0	0.7

Next, Lund's IRDF technique, adapted for the discrete-time case as discussed in Section 3.6, is implemented in the MMAE with the tuning values shown in Table 2. These tuning values lead to enhanced parameter estimation performance with minimal impact on the M^{3}AE's state estimation performance, as will be shown later in this section.

Figures 15 - 18 compare the outputs from a 10-run Monte Carlo simulation of the MMAE (tuned for optimized parameter estimation, and with $\mathbf{a}_\mathrm{T} = 32.0$) before and after Lund's discrete-time IRDF is applied. The top plots in Figures 15 and 16 present the parameter estimate performance (dotted line) versus the true parameter value (the solid line) for a representative sample run from the 10-run Monte Carlo simulation. The bottom plots represent the mean error (dotted line) and the mean ± 1 standard deviation (dashed line) values from the 10-run Monte Carlo simulation, where the error is defined as:

$$\mathbf{e}_a(t_i) = \hat{\mathbf{a}}_{MMAE}(t_i) - \mathbf{a}_\mathrm{T}(t_i) \tag{175}$$

Notice that both plots in Figure 15 indicate that the MMAE-supplied parameter estimate converges to the $\mathbf{a}_j$ *closest* to $\mathbf{a}_\mathrm{T}$ in the "Baram distance measure sense," namely $\omega_n = 28.40$ rad/sec. The

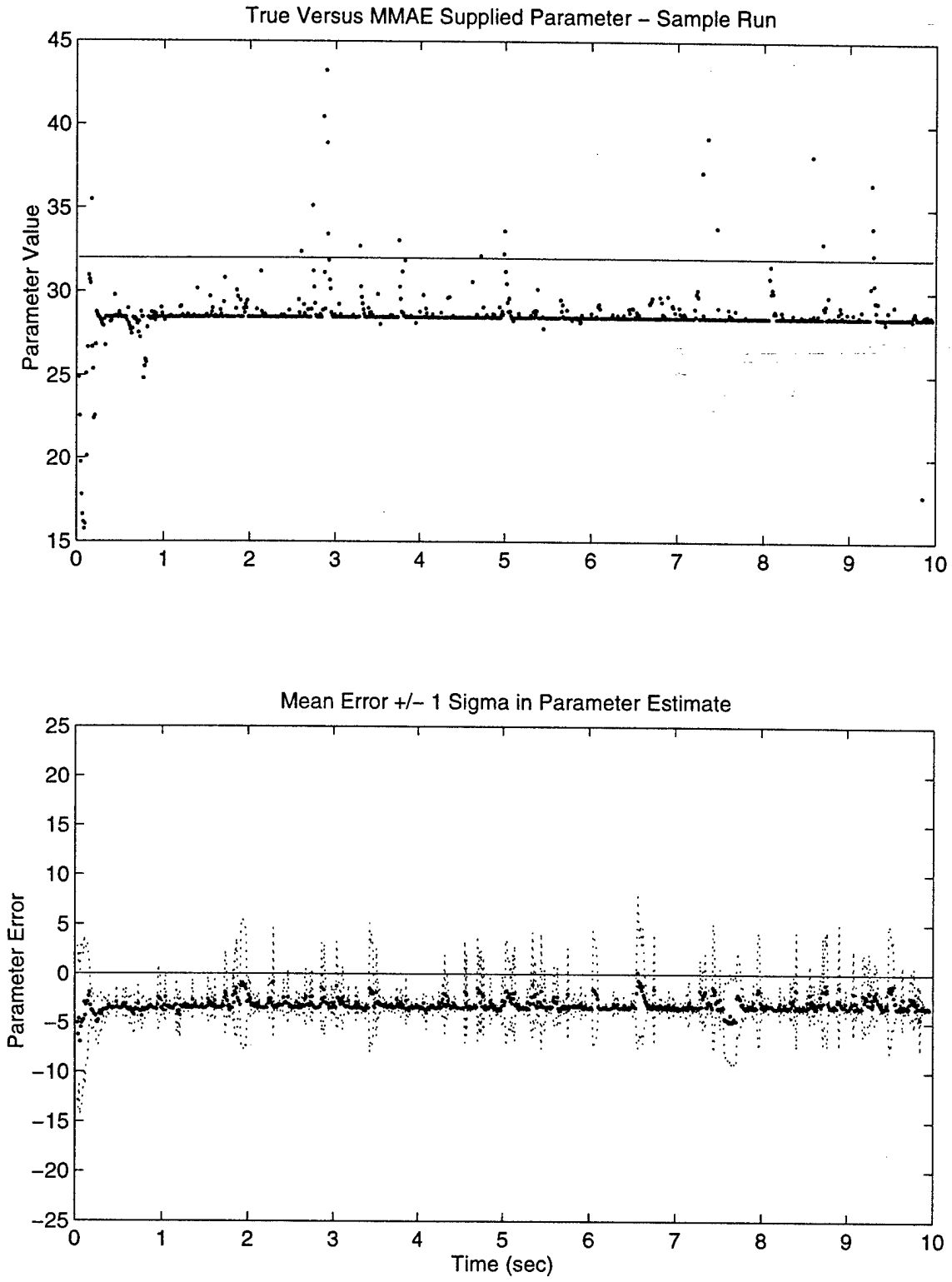


Figure 15. *MMAE/SDPEP* Without IRDF Parameter Estimation Performance: Test Case 1

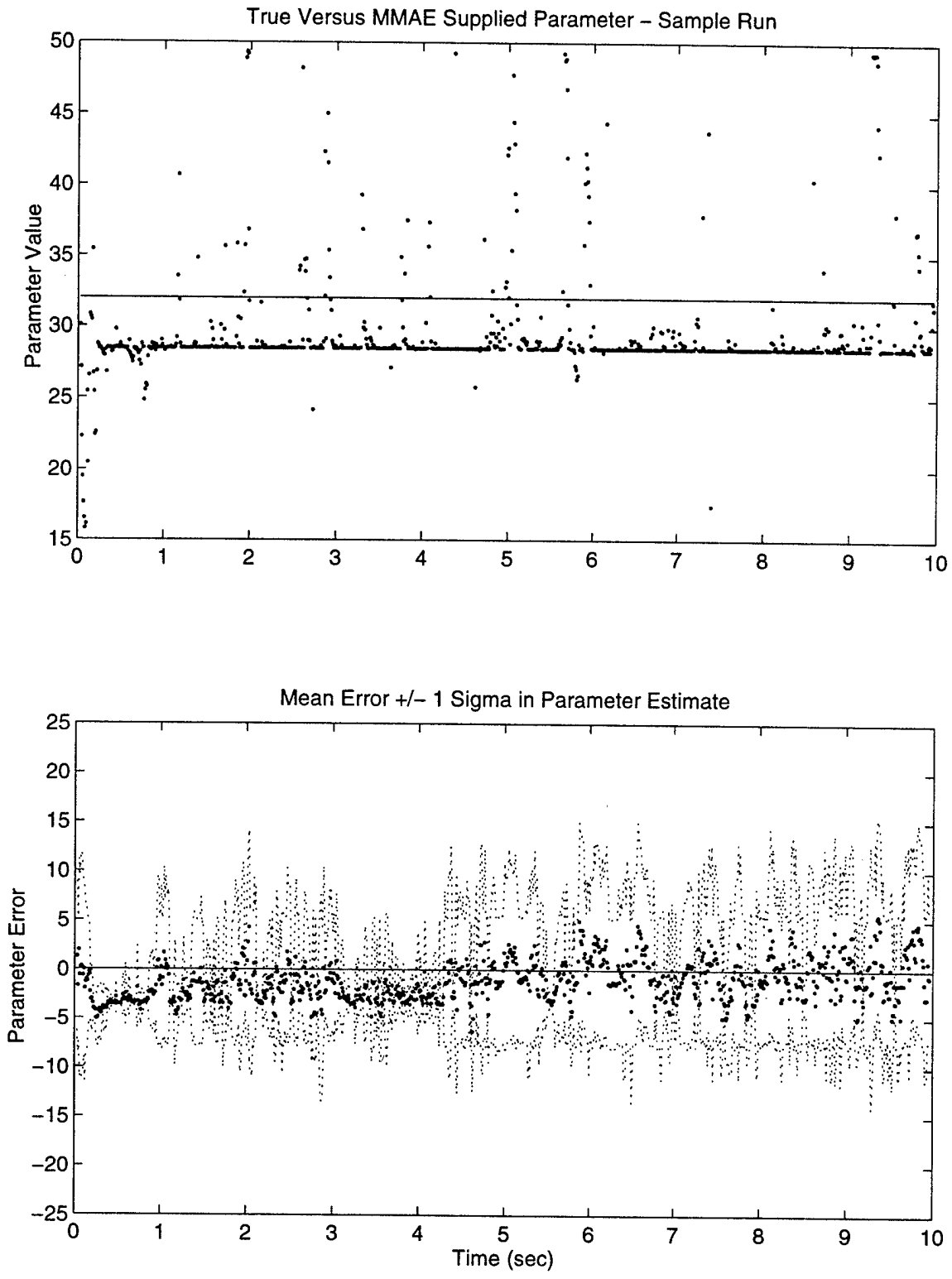


Figure 16. *MMAE/SDPEP* With IRDF Parameter Estimation Performance: Test Case 1

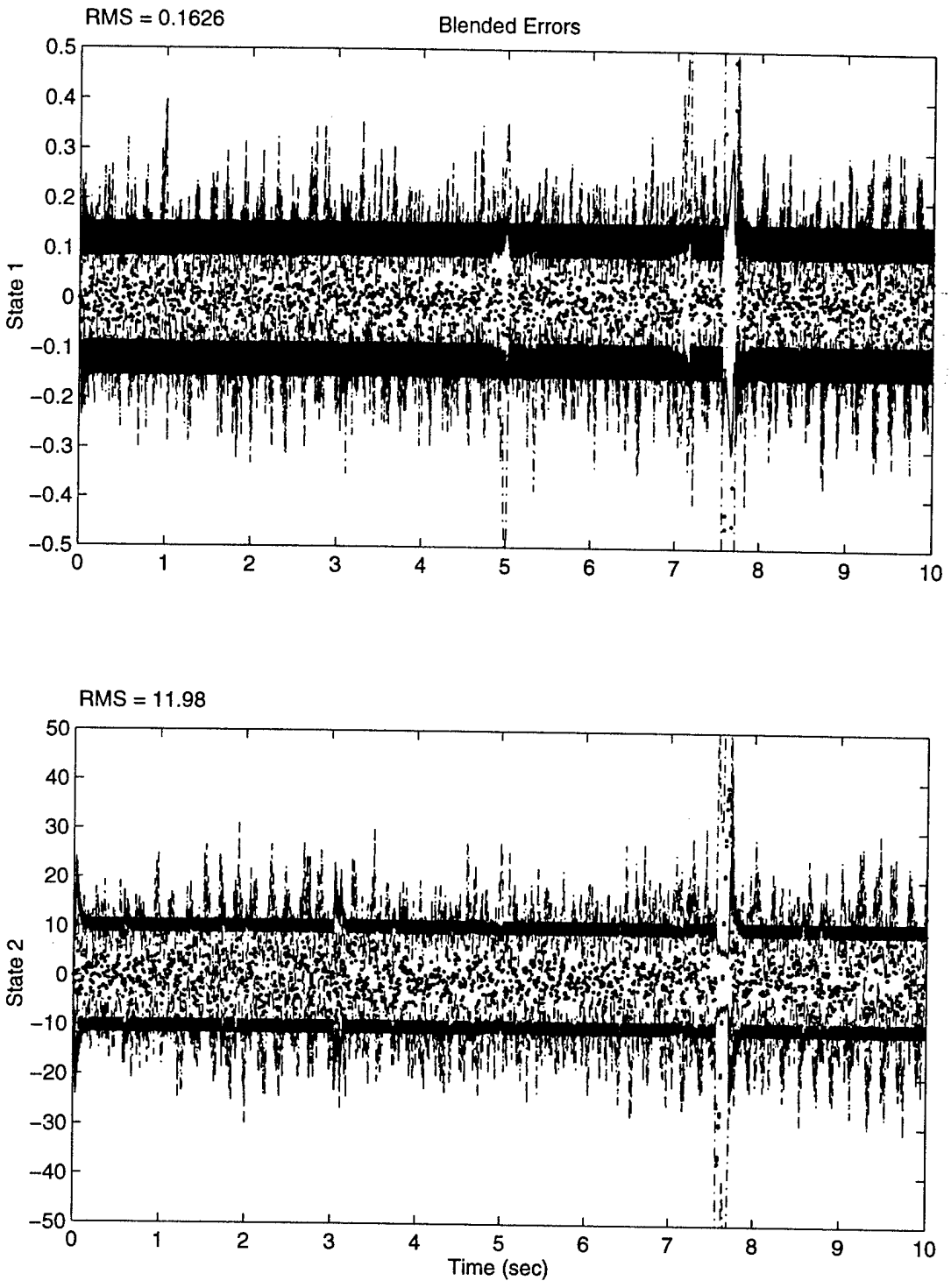


Figure 17. *MMAE/SDPEP* Without IRDF State Estimation Performance: Test Case 1

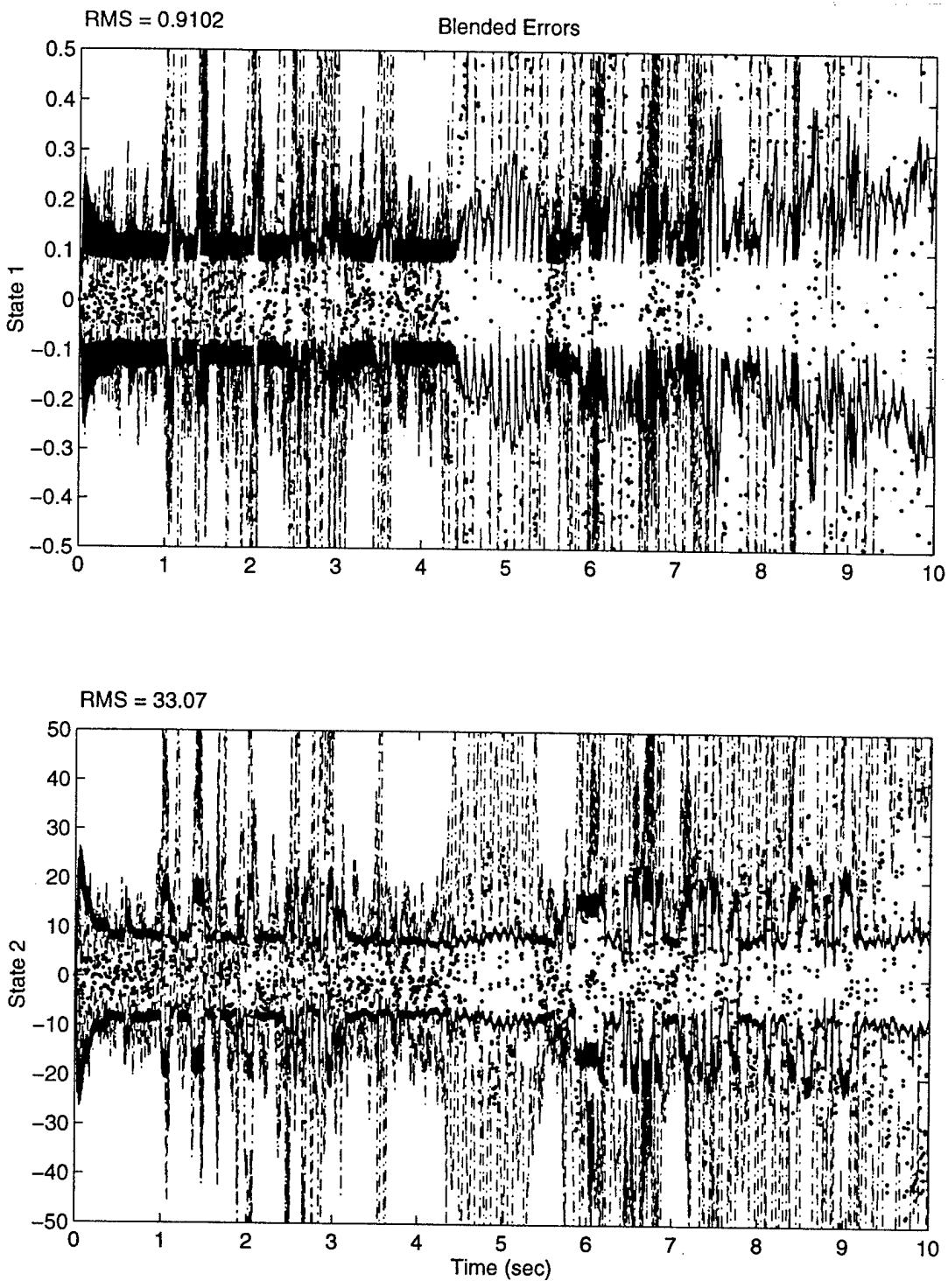


Figure 18. *MMAE/SDPEP* With IRDF State Estimation Performance: Test Case 1

plots indicate little blending, thus the resulting bias in the estimate. The plots in Figure 16 indicate a change once IRDF is incorporated. The top plot indicates the same trends as the top plot in Figure 15, but this is only a single-run time history. Notice the parameter estimates in the bottom plot (10 run Monte Carlo simulation). The resulting parameter estimate is a close to a zero-mean estimate, indicating a more accurate parameter estimate for input into the M^3AE's state estimator, on average.

Figure 18 shows the impact to the state estimation performance after implementing IRDF, as compared to Figure 17. Note that this is the blended *state* estimate from the MMAE designed for best *parameter* estimation. These plots are shown to emphasize the negative impact to an MMAE's state estimation performance when implementing IRDF, even though this configuration would not typically be used for state estimation. The top plots in Figure 17 and 18 indicate the MMAE's blended state x_1 estimation performance, whereas the bottom plots indicate the MMAE's blended state x_2 estimation performance. Notice that the scales on *all* such state estimation performance plots are the same for easy comparison. Each plot contains 5 curves representing the following:

- The solid trace represents the zero (assumed mean) $\pm$ one filter-predicted sigma bound for the error in the blended estimate of each state variable.
- The dotted line represents the actual 10-run mean value of the error for each state variable.
- The dashed line represents the 10-run mean $\pm$ one sigma error for each state variable.

Additionally, to assist in analyzing the state plots, a performance measure representing the temporally averaged root-mean-square ($\overline{RMS}(e)$) of the state estimation error, $e_\omega(t_i) = \hat{x}_\omega(t_i) - x_{true_\omega}(t_i)$, is given by:

$$\overline{RMS}(e) = \frac{1}{N_{samp}} \sum_{i=0}^{N_{samp}} \sqrt{\frac{1}{N_{runs}} \sum_{\omega=1}^{N_{runs}} e_\omega^2(t_i)} \tag{176}$$

where:

$\hat{x}_\omega(t_i)$	=	filter-computed estimate of a given state for a given sample (ω) of the stochastic process
$x_{true_\omega}(t_i)$	=	*truth* model value of the same state for that sample (ω) of the process
N_{runs}	=	number of time histories in the Monte Carlo simulation
ω	=	single time history run number
N_{samp}	=	the number of sample periods in each time history

These measure values pertain to errors at both t_i^+ and t_i^- and are indicated in the upper left of each plot with "RMS =...".

Notice how the *MMAE/SDPEP* 's state estimation performance suffers with IRDF, as evident from Figure 18 and the increase in the temporally averaged RMS state estimation errors associated with the IRDF case shown in Table 3 below.

Table 3. IRDF's Temporally Averged RMS State Estimation Errors

State	Before IRDF	IRDF	Percent Increase (%)
x_1	0.1626	0.9102	459.8
x_2	11.98	33.07	176.0

Notice however, as mentioned above, that the IRDF-based MMAE's parameter estimate has a mean error closer to zero than does the non-IRDF-based MMAE, indicating a better estimate of the true parameter value. This confirms the discussion in Section 2.3 concerning the trade-off between state and parameter estimation performance, and this example clearly indicates the benefit to be gained with the M^{3}AE architecture (discussed in more detail in Section 4.1.3.1). The M^{3}AE's state estimation performance, given $\hat{\mathbf{a}}$ from the IRDF-based MMAE, is generally just as good as the state estimation performance produced by the MMAE "Sheldon-discretized" for optimized state estimation. Where significant blending occurs in the estimation of $\hat{\mathbf{a}}$, the estimate itself is more representative of the true parameters than any of the $\mathbf{a}_j$ values used for the basis of the elemental filters, and the M^{3}AE single filter state estimate can then substantially outperform the precision achievable with a

conventional MMAE. However, without MMAE blending of the parameter estimates, the M^{3}AE's state estimation performance provides no clear advantage over existing MMAE techniques.

The effects of IRDF are further emphasized after reconducting the above test case with $\eta(t_i)$ artificially set to zero, rather than just letting $\eta_{\min}$ be zero, causing $\mathbf{Q}'_{\mathrm{d}j}(t_i) = \eta(t_i)\mathbf{Q}_{\mathrm{d}j}(t_i) = \mathbf{0}$ throughout the simulation. This is not a practical test in determining system performance, since setting the $\mathbf{Q}'_{\mathrm{d}j}(t_i)$'s to zero has the effect of reducing the elemental filter gains below anything reasonable compared to that which would be obtained via conventional tuning methods. As a result, the MMAE state estimation performance suffers greatly, as evident from Figure 19, which plots the MMAE's blended state estimation performance. The associated RMS values of the state estimation errors are shown in the Table 4 below. Yet, even in this extreme case, the parameter estimate performance shown in Figure 20 is comparable to that of the MMAE without IRDF implemented, as shown in Figure 15. Moreover, of greater significance, the corresponding M^{3}AE's state estimation performance is "comparable" to the *MMAE/SDSEP* 's state estimation performance, as will be shown in Section 4.1.3.

Table 4. IRDF's Temporally Averged RMS State Estimation Errors ($\eta(t_i) = 0$)

State	Before IRDF	IRDF	Percent Increase (%)
x_1	0.1626	1.889	1061.7
x_2	11.98	60.55	405.4

However, the negative impact of intentionally setting $\eta(t_i)$ artificially to zero, is evident upon closer inspection of the top plot in Figure 20 where, during the majority of the simulation, the $\hat{\mathbf{a}}$ is equal to $\frac{1}{3}\sum \mathbf{a}_j$'s from the 3 MMAE elemental filters (the obvious value of 30.58, the average of 13.44, 28.40, and 49.90 from Table 1, all of which are also separately evident in the top plot of Figure 20). This is evident upon inspection of Figure 21 which indicates the elemental filter probabilities

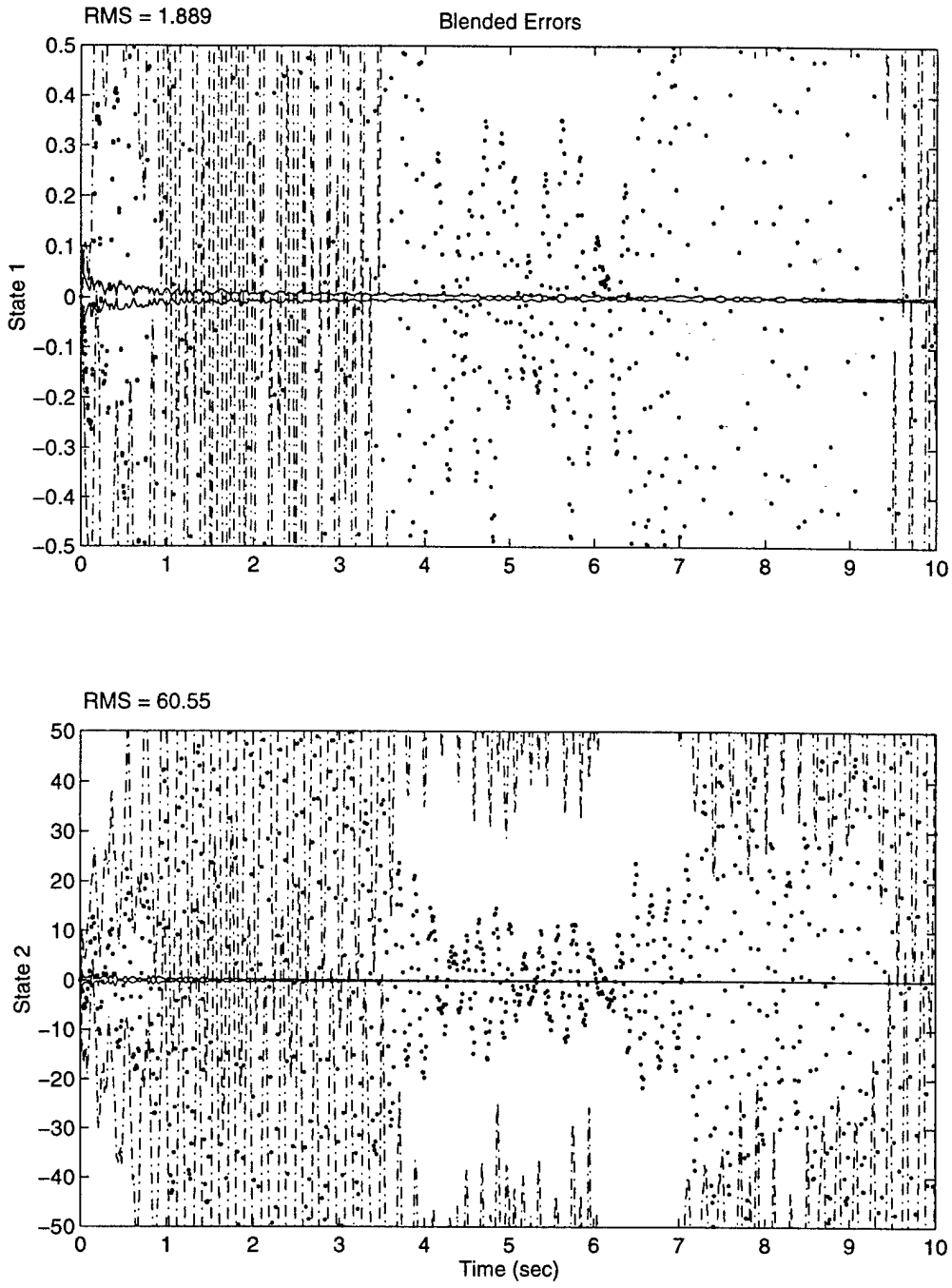


Figure 19. *MMAE/SDPEP* With IRDF State Estimation Performance: Test Case 1, $\eta(t_i)$=0

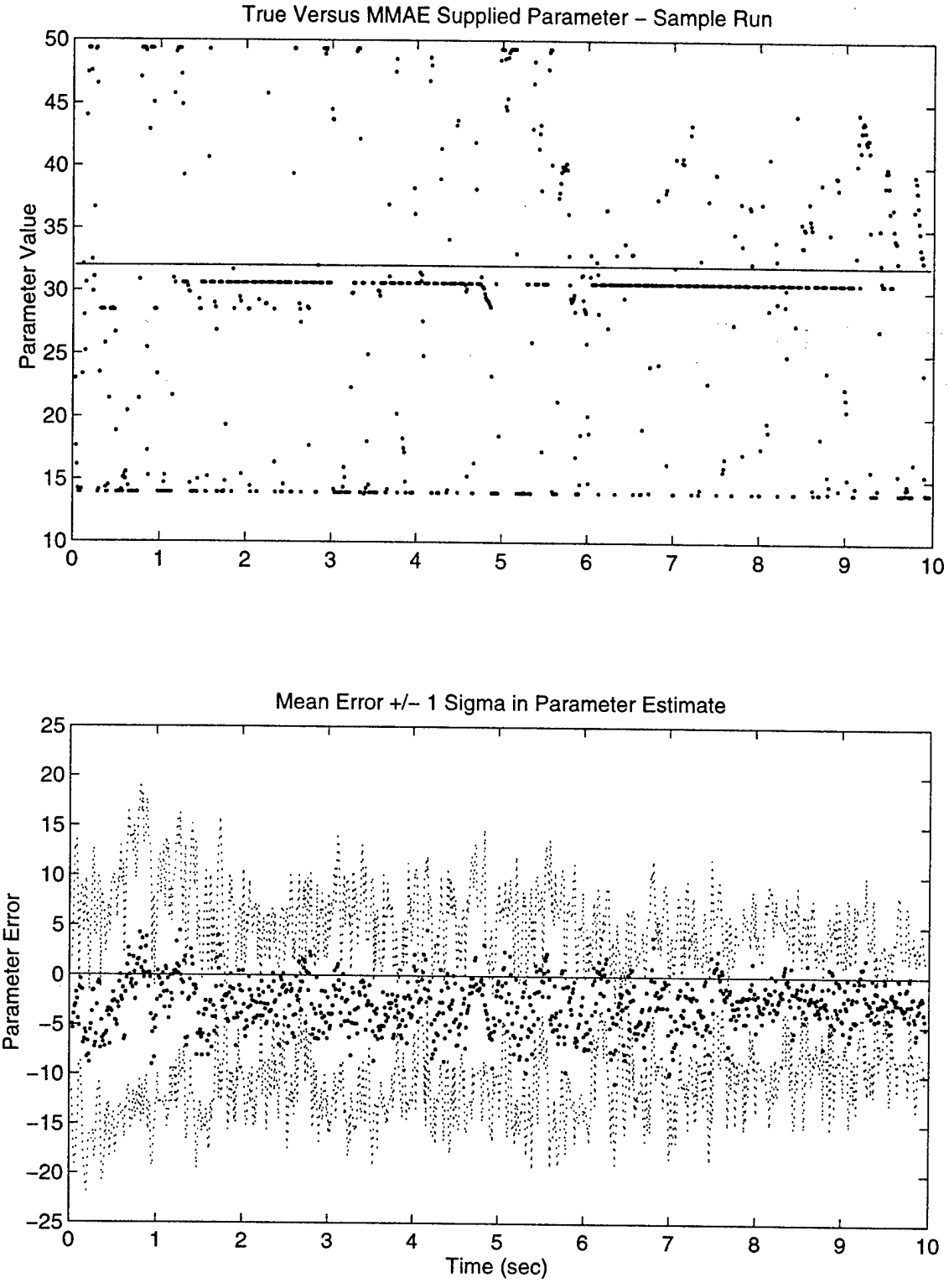


Figure 20. *MMAE/SDPEP* With IRDF Parameter Estimation Performance: Test Case 1, $\eta(t_i)$=0

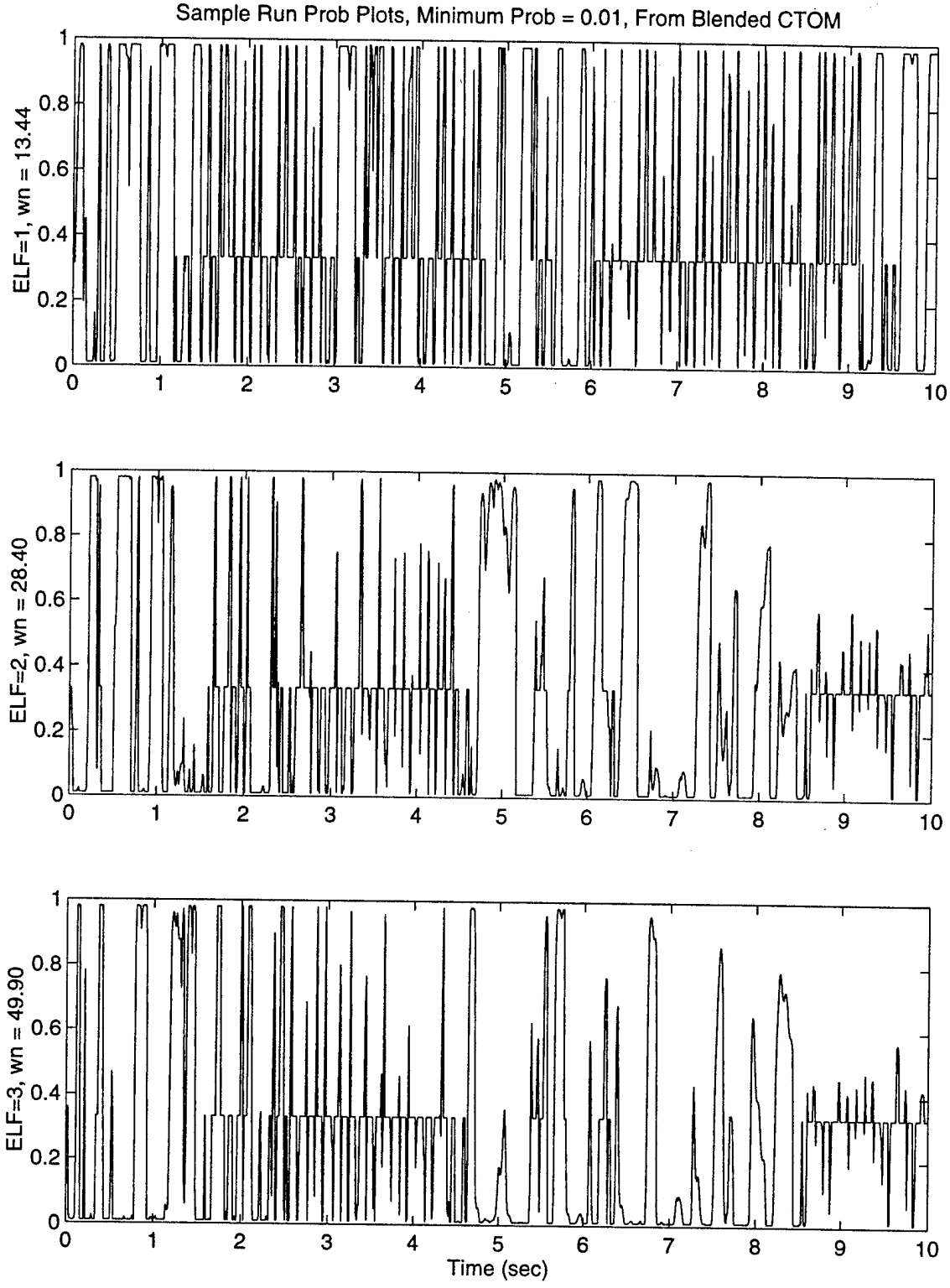


Figure 21. *MMAE/SDPEP* With IRDF Probability Plots: Test Case 1, $\eta(t_i)$=0

associated with that sample run. Obviously, the elemental filters are sharing the probability equally during the majority of the run, indicating the MMAE has become "lost" since none of the "detuned" elemental filters are providing adequate estimates. All elemental estimators have residuals that look equally bad, and so a probability of $\frac{1}{3}$ is assigned to each. Thus, the main reason good parameter estimation occurs is that the $\frac{1}{3}\sum \mathbf{a}_j = 30.58$ parameter estimate value *happens to be* very *close* to $\mathbf{a}_T = 32.0$ in this particular example, and this should not be misinterpreted as effective performance by the estimator.

4.1.2.3 M^{3}AE Covariance Analysis

The M^{3}AE approximate covariance analysis tool is implemented in MATLAB. This section provides the results from conducting the M^{3}AE approximate covariance analysis. Eight test cases are conducted to compare performance between the M^{3}AE architecture and a conventional fixed-bank *MMAE/SDSEP*. The eight test cases are listed in Table 5.

The first seven test cases keep the true value of the ω_n parameter $\mathbf{a}_T$, fixed throughout the simulations. Test case 1 is chosen as a baseline to compare to the results from Sheldon's dissertation [69]. Test cases 2 and 5 set $\mathbf{a}_T$ equal to the same value as the second elemental filter's parameter value, $\mathbf{a}_2$, in the *MMAE/SDSEP* and *MMAE/SDPEPs* respectively. Test cases 3, 4, 6, and 7 place the true parameter value midway between the two adjacent elemental filters' $\mathbf{a}_j$'s, for each set of parameter discretization values (see the comments in Table 5), so that no single elemental filter has the correctly assumed parameter value. Finally, test case 8 investigates the M^{3}AE's ability to handle a small step change in the true parameter value from 35.0 to 30.385 at the $t_i = 3$ second point. These values were chosen due to their proximity to the *MMAE/SDSEP* and *MMAE/SDPEP* second elemental filter's $\mathbf{a}_2$ values.

Table 5. M^3AE Test Cases

Test Case	$\omega_n = \mathbf{a}_T$	Comments
1	32.00	Sheldon example (all cases have $p_{min} = 0.01$)
2	37.89	$\mathbf{a}_2 = \mathbf{a}_T$ for State Optimized $\mathbf{a}$: 22.88 37.89 54.43
3	30.385	Midpoint between first 2 elemental filters: $\frac{\mathbf{a}_1+\mathbf{a}_2}{2}$
4	46.16	Midpoint between last 2 elemental filters: $\frac{\mathbf{a}_2+\mathbf{a}_3}{2}$
5	28.40	$\mathbf{a}_2 = \mathbf{a}_T$ for Parameter Optimized $\mathbf{a}$: 13.44 28.40 49.90
6	20.92	Midpoint between first 2 elemental filters: $\frac{\mathbf{a}_1+\mathbf{a}_2}{2}$
7	39.15	Midpoint between last 2 elemental filters: $\frac{\mathbf{a}_2+\mathbf{a}_3}{2}$
8	varies	Step change in $\mathbf{a}_T$ from 35.0 to 30.385 at $t_i = 3$ seconds

A one-run Monte Carlo simulation is conducted on the M^3AE's MMAE *only*, generating the time histories for the parameter estimates $\hat{\mathbf{a}}(t_i)$, the filter-computed covariance $\mathbf{P}_{\mathbf{a}}(t_i)$, and the associated elemental filter probabilities $p_j(t_i)$ for each test case. These time histories are then supplied to the M^3AE approximate covariance analysis tool. Four representative test cases (1, 2, 5, and 8) are discussed in this section. The remaining four test cases (3, 4, 6, and 7) provided no additional insight into the M^3AE performance and thus are not presented. Figures 22-25 present the results from test cases 1, 2, 5, and 8. These plots are then compared to the actual results obtained from each test case for which a complete 10-run Monte Carlo simulation of the M^3AE is conducted; see Figures 27, 36, 34, and 43 on pages 128, 142, 138, and 152, respectively.

In each case, the M^3AE approximate covariance analysis tool performs well in predicting the expected performance of the M^3AE, as compared to the actual results from each 10-run Monte Carlo simulation of the M^3AE. In general, however, predictions associated with state x_1 are more accurate than state x_2, especially after each measurement update. This is directly attributable to the problem setup. Notice that in Equation (174), a *direct* measurement of the position is available at each sample time, whereas no such *direct* measurement is available for the velocity state. Additionally, the "further" away $\mathbf{a}_T$ is from the *closest* MMAE's elemental filter parameter value, $\mathbf{a}_j$, the larger the

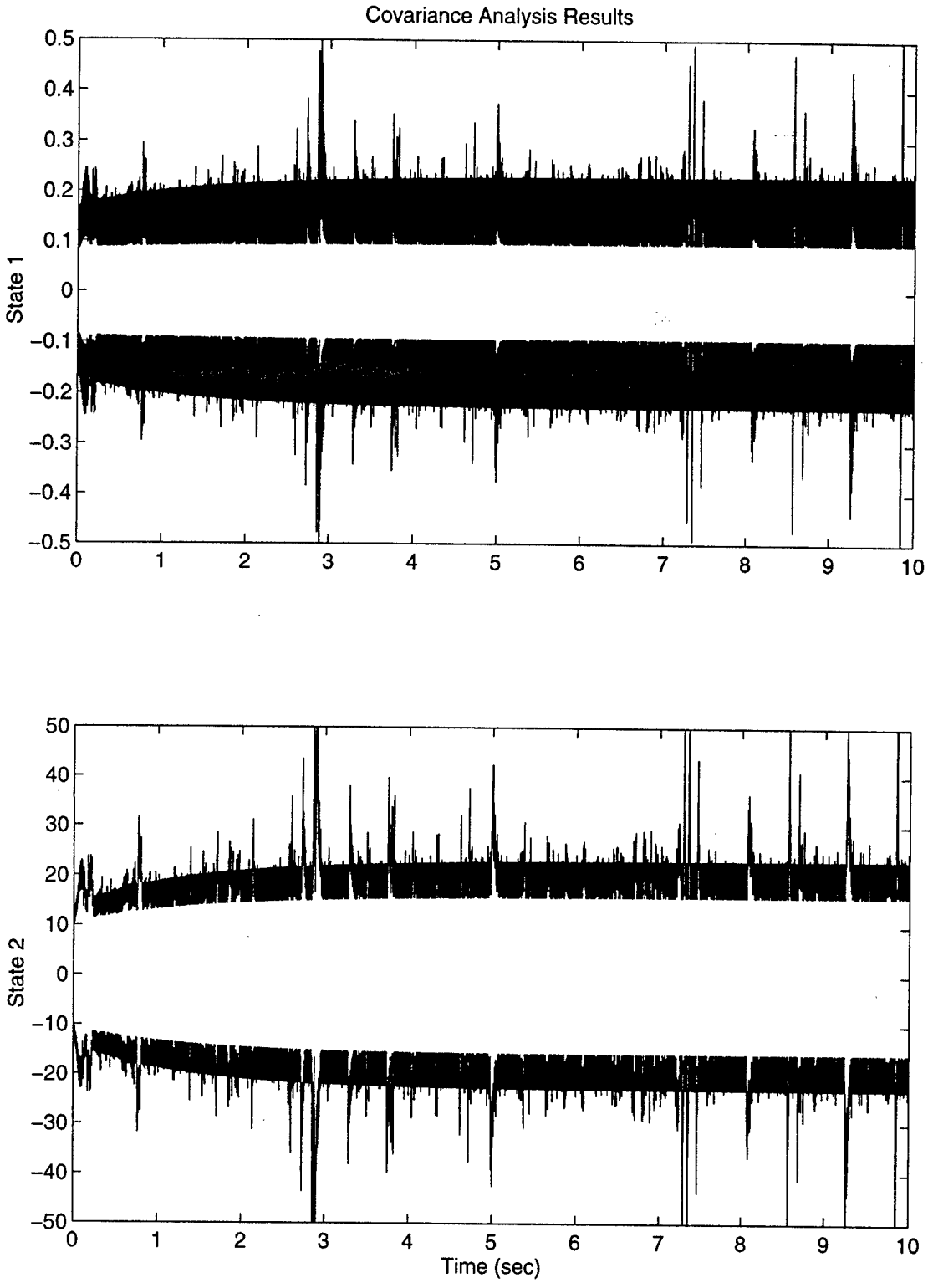


Figure 22. M^{3}AE Covariance Analysis: Test Case 1

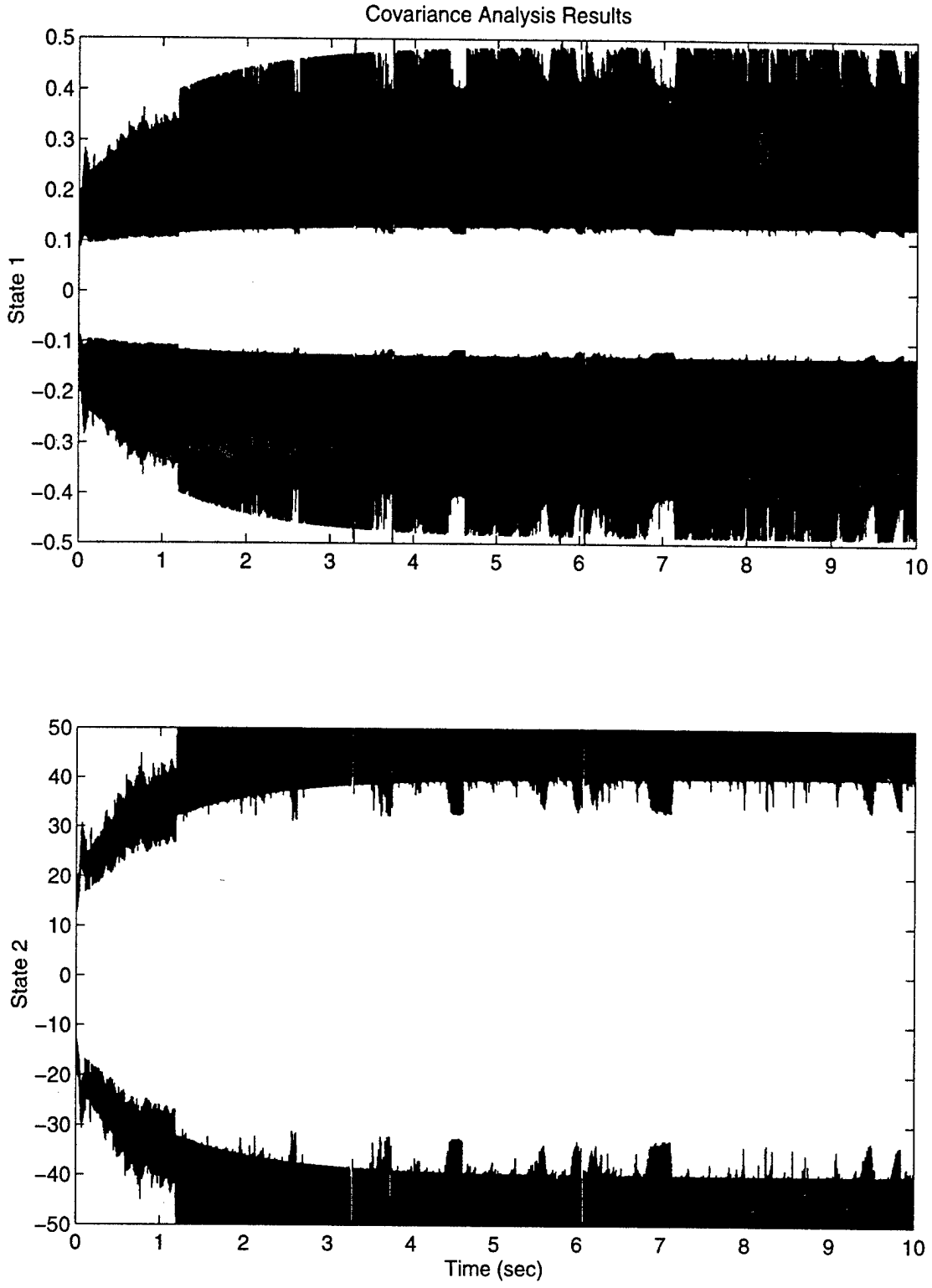


Figure 23. M^3AE Covariance Analysis: Test Case 2

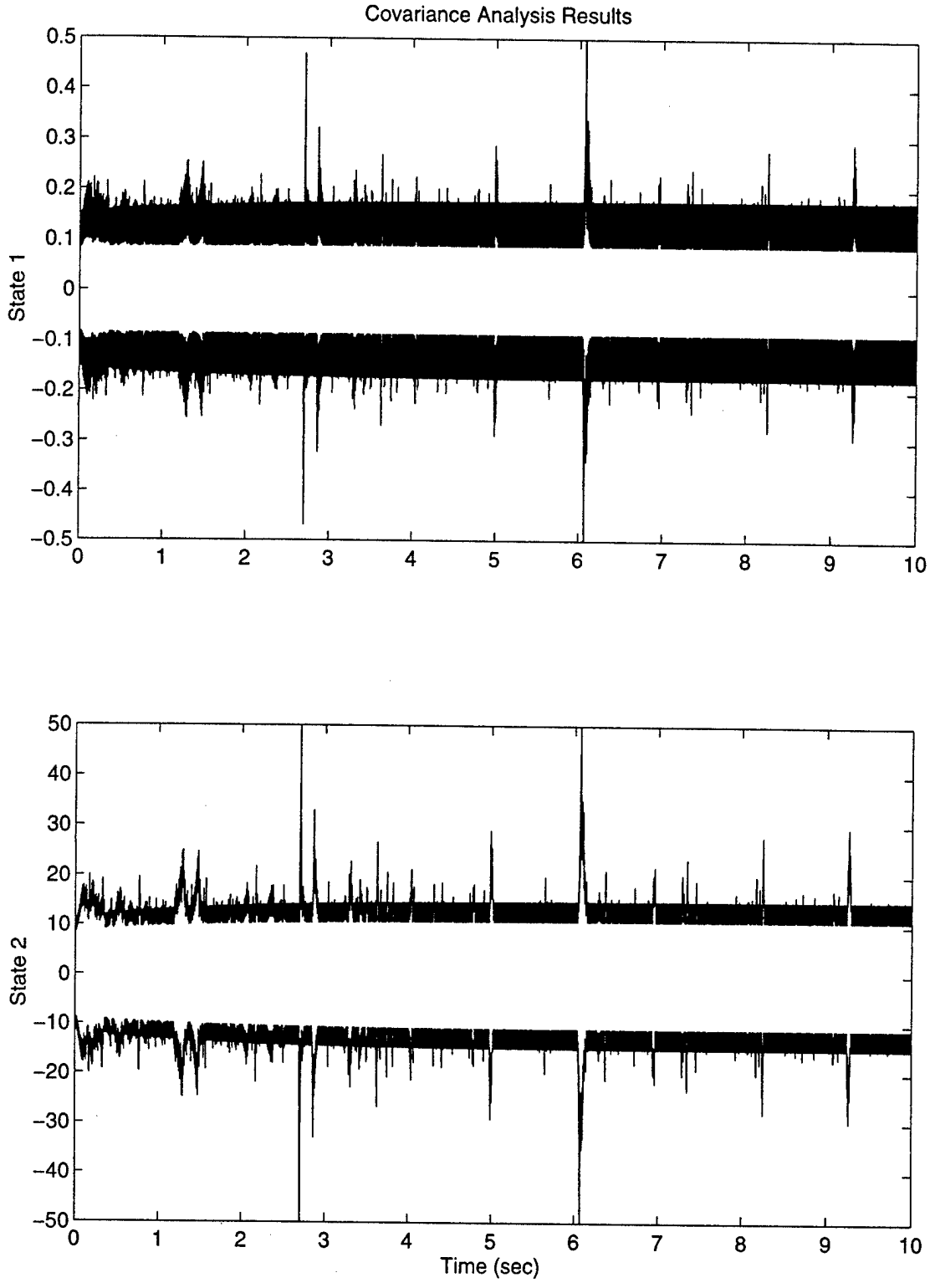


Figure 24. M^{3}AE Covariance Analysis: Test Case 5

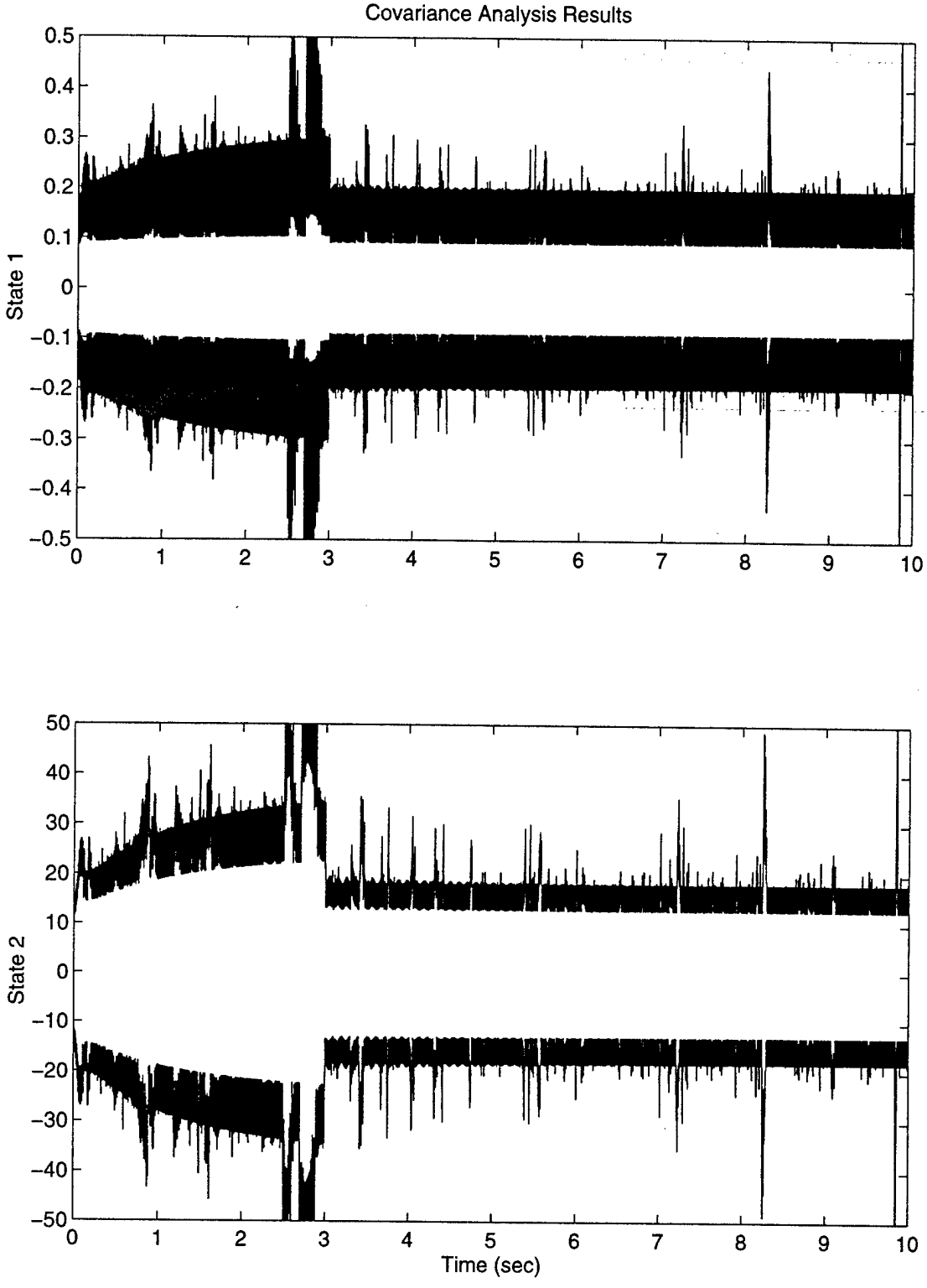


Figure 25. M^{3}AE Covariance Analysis: Test Case 8

bias in the parameter estimate, $\hat{\mathbf{a}}_{MMAE}$. This larger bias directly impacts the M^{3}AE approximate covariance analysis tool since the first order error term in Equation (105) will start to dominate, versus the situation when the bias in the MMAE supplied parameter estimate $\hat{\mathbf{a}}_{MMAE}$ is small, in which case the zero order term dominates. This is evident upon inspection of the M^{3}AE approximate covariance plots and the parameter estimate plots. As shown in test cases 1, 5, and 8, the approximate covariance prediction error is small when the true parameter value $\mathbf{a}_T$ is *close* to the blended parameter estimate $\hat{\mathbf{a}}_{MMAE}$, and grows larger as the true parameter value $\mathbf{a}_T$ moves away from the blended parameter estimate $\hat{\mathbf{a}}_{MMAE}$, as in test case 2. This is expected and is indicative of the degradation in performance of an estimator based on an incorrect or "biased" parameter value. If this "bias-like" or "mean-like" term becomes the predominate portion of $\Psi = \mathbf{P} + \mathbf{m}\mathbf{m}^T$, then the technique discussed in Section 3.3.1.3 could be implemented to account for the cases in which $\mathbf{a}_T$ is *far* from the *nearest* $\mathbf{a}_j$. This case strongly motivates trying to reduce the bias in the estimation of parameters, and specifically Chapter 5 will recommend extending research to *moving-bank* forms of MMAE and M^{3}AE to accomplish this purpose.

4.1.3 Simulations and Performance Analysis

This section summarizes the results from the test cases described in Section 4.1.2.3. In each test case, a 10-run Monte Carlo simulation is conducted and a comparative analysis is accomplished between the following three architectures:

1. An MMAE "Sheldon-discretized" for optimized state estimation performance (*MMAE/SDSEP*)
2. An M^{3}AE implemented without IRDF (with internal *MMAE/SDPEP*)
3. An M^{3}AE implemented with discrete-time IRDF (with internal *MMAE/SDPEP*)

Thus, (1) and the MMAE within (2) represent conventional MMAEs, optimally discretized for state or parameter estimation performance, respectively. The single state estimator within the M^{3}AE in (2) can be compared to the blended state estimate from either (1) or (2), and this will be done in the following discussion. Finally, (3) will show the impact of IRDF, as compared to (2) and to the conventional MMAE's of (1) and (2).

Even though eight test cases were conducted, only the four test cases mentioned earlier are examined in detail in this section. Several types of plots are presented upon which the analysis is based. The state estimation performance plots presented are:

1. The *MMAE/SDSEP* 's blended output state estimation performance (10-run Monte Carlo simulation) for states x_1 and x_2.

2. The M^{3}AE's *MMAE/SDPEP* (designed without IRDF) blended output state estimation performance (10-run Monte Carlo simulation) for states x_1 and x_2.

3. The M^{3}AE's *MMAE/SDPEP* /IRDF (designed with IRDF) blended output state estimation performance (10-run Monte Carlo simulation) for states x_1 and x_2.

4. The M^{3}AE (designed without IRDF) single-filter state estimation performance (10-run Monte Carlo simulation) for states x_1 and x_2.

5. The M^{3}AE/IRDF (designed with IRDF) single-filter state estimation performance (10-run Monte Carlo simulation) for states x_1 and x_2.

Note that the *MMAE/SDSEP* /IRDF (designed with IRDF) is not investigated in this effort, since its purpose in the M^{3}AE architecture is to enhance the MMAE-based parameter estimation performance. Additionally, state estimation performance *typically* suffers when IRDF is implemented, as will be shown in the results presented later in this chapter. The intended comparisons to be accom-

plished between the various state estimation performance plots are as follows: plots 4 $\Leftrightarrow$ 2 $\Leftrightarrow$ 1, and 5 $\Leftrightarrow$ 4, and 3 $\Leftrightarrow$ 2, while *simultaneously* considering parameter plots (listed below) for *all* the cases.

In addition to the state estimation performance plots, the parameter estimation performance plots are considered simultaneously. The plots presented are:

1. A representative sample plot from each architecture for the MMAE's blended parameter estimate performance versus a_T (one-run Monte Carlo simulation).

2. The MMAE's parameter estimation performance (10-run Monte Carlo simulation).

Table 6 summarizes the plots generated for each test case versus the three architectures listed above. The figure numbers associated with each case are listed in the columns.

Table 6. Summary of Plots

Architecture	MMAE Blended State Est.	M^3AE State Est.	Parameter Estimation
M^3AE without IRDF	17, 36, 31, 40	27, 34, 43	15, 37, 45
M^3AE with IRDF	18, 19, 32, 41	28, 29, 35, 44	16, 20, 38, 46
MMAE/SDSEP	26, 33, 42		30, 39, 47

Besides the first test case, the basic flow of the figures is as follows: three figures depicting the blended MMAE state estimation performance; two figures depicting both M^3AEs (without and with IRDF) state estimation performance; and finally, three figures depicting the MMAEs parameter estimation performance. In addition to the plots provided for each test case, a table summarizes the temporally averaged RMS values of the state estimation errors. However, extreme care should be taken when reviewing these values. Short regions of high-variance data tend to increase the

temporally averaged RMS value of the state estimation errors for a given plot. Thus, the plots should be used as the primary indicator of performance, whereas the RMS values provide only a gross indication of state estimation performance, since any amount of high-variance data may adversely skew the results.

4.1.3.1 Test Case 1: $a_T = 32.0$

This test case was selected to serve as a baseline during the M^{3}AE development. It is the same test case presented in Sheldon's research example [69]. An analysis of the actual state and parameter estimation performance is presented. First the state estimation performance will be analyzed, followed by the parameter estimation.

State Estimation Performance. Figures 17, 18 (see pages 107 and 108), and 26 show the plots from each MMAE's blended state estimation performance. The temporally averaged RMS state estimation errors from each plot are summarized in Table 7 below.

Table 7. Test Case 1: MMAEs' Temporally Averged RMS State Estimation Errors

State	*MMAE/SDPEP* without IRDF	*MMAE/SDPEP* with IRDF	Conventional *MMAE/SDSEP*
x_1	0.1626	0.9102	0.3986
x_2	11.98	33.07	17.22

Notice that, even without the M^{3}AE implemented in this case, the *MMAE/SDPEP* (without IRDF) outperforms the conventional *MMAE/SDSEP*. This is opposite to the scenario discussed earlier in Section 4.1.2.1. In this test case, a_T is *closer* to an elemental filter's a_j in the M^{3}AE's *MMAE/SDPEP* than to any elemental filter's a_j in the *MMAE/SDSEP*. Thus, not only is the parameter estimation performance better (as shown in the *Parameter Estimation* subsection below), but the state estimation performance is also better. Additionally, note that the *MMAE/SDPEP* /IRDF's

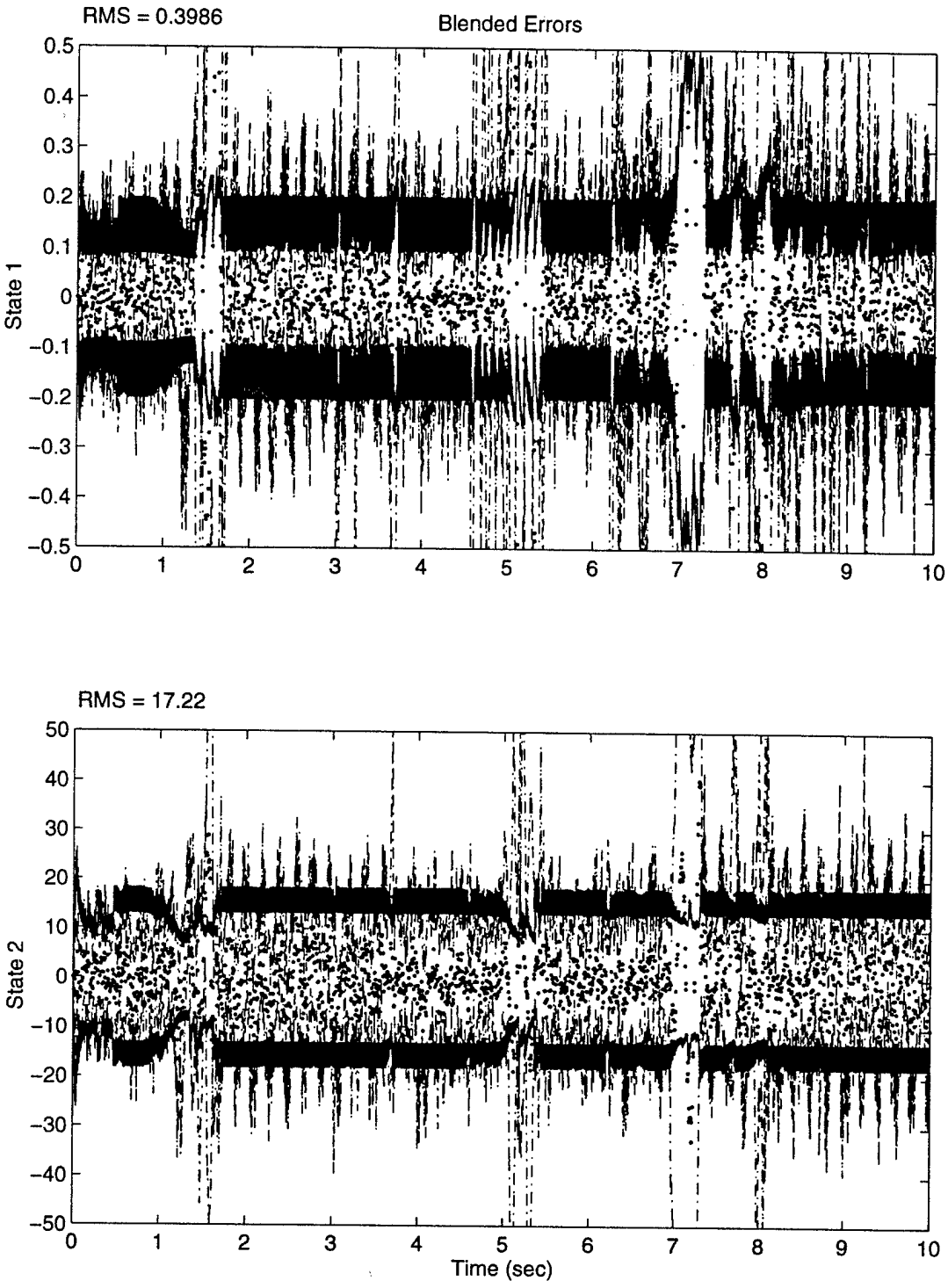


Figure 26. *MMAE/SDSEP* State Estimation Performance: Test Case 1

results further emphasize the negative impact of applying IRDF to an MMAE if its intended purpose is to provide accurate state estimation. For example, Figure 18 on page 108 (compared to Figure 17 on page 107) clearly indicates the deterioration in the *MMAE/SDPEP* /IRDF's blended state estimate performance, starting at approximately 4.2 seconds.

Furthermore, notice in Figures 17 and 15, at approximately 7.7 seconds, a small section of high-variance data representing degradation in both the state and parameter estimation performance. Upon closer inspection of Figure 17, the filter-predicted standard deviation in x_2 errors has decreased inappropriately, thus the MMAE "thinks" performance has improved even though it has not. Therefore, the MMAE is underestimating its own errors and the resulting true errors grow substantially larger than they are at the other times as well.

Next, note that the M^{3}AE state estimation performance with or without IRDF is better than the conventional *MMAE/SDSEP*, as shown in Figures 27 - 29 (as compared to Figures 17 - 19, respectively). The temporally averaged RMS state estimation errors from each plot are summarized in the Table 8. Furthermore, notice the significant improvement in the M^{3}AE/IRDF's state estimation performance as compared to the conventional MMAE's (implemented with IRDF) state estimation performance. In fact, it performs almost as well in state estimation performance as the M^{3}AE implemented without IRDF. This performance improvement results from providing accurate blended parameter estimates to the single Kalman filter in the M^{3}AE architecture, as discussed in the next section. Clearly, all of the M^{3}AEs outperform the *MMAE/SDSEP*. Also, notice that the greatest improvement in state error performance of the M^{3}AE over the *MMAE/SDSEP* occurs in state x_1 at the update cycle, for the reasons expressed at the end of Section 4.1.2.3 – i.e., a *direct* measurement of the position is available at each sample time providing an accurate measurement update. Direct measurements of x_1 (that are reasonably precise) should cause x_1 estimation errors to be *insensitive* to bad parameter estimates, whereas x_2 estimates (with velocity not measured *directly*) should be

more *sensitive* to bad parameter estimates. However, as shown in Figures 27 and 28, state x_2 experiences an improvement in its state estimation performance and its $\pm$ one filter-predicted-sigma performance (the standard deviation of the estimate error), as compared to Figures 17 and 18. This is anticipated and is a direct result of using a more accurate parameter estimate (than any of the MMAE's $\mathbf{a}_j$ values) in the single Kalman filter state estimator. Finally, as shown in Figure 29 and Table 8, there is *significant* improvement in the M^{3}AE's state estimation performance relative to the *MMAE/SDPEP* implemented with IRDF and $\eta(t_i) = 0$. This is a direct result of providing the M^{3}AE's single Kalman filter a decent blended parameter estimate throughout the simulation, as shown in Figure 20 on page 113.

Table 8. Test Case 1: M^{3}AEs' Temporally Averged RMS State Estimation Errors

State	M^{3}AE without IRDF	M^{3}AE with IRDF	M^{3}AE with IRDF, $\eta = 0$
x_1	0.1256	0.1379	0.2850
x_2	10.96	13.70	14.58

Parameter Estimation Performance. Figures 15, 16, 20 (see pages 105, 106, and 113), and 30 show the plots of each MMAE's blended parameter estimation performance. As mentioned earlier in Section 4.1.2.2, the plots in Figures 15 and 16 indicate two different levels of parameter estimation performance, with IRDF yielding considerable benefit. Notice the mean errors in the parameter estimates shown in the bottom plot in each figure. There exist distinct biases in the parameter estimates from the *MMAE/SDSEP* (notice the plots in Figure 30, which indicate that the majority of the probability weight is given to the *MMAE/SDSEP* 's second elemental filter's, $\mathbf{a}_2 = 37.89$, which is *closest* to $\mathbf{a}_T$ in the "Baram distance measure sense," resulting in a bias in the parameter estimate),

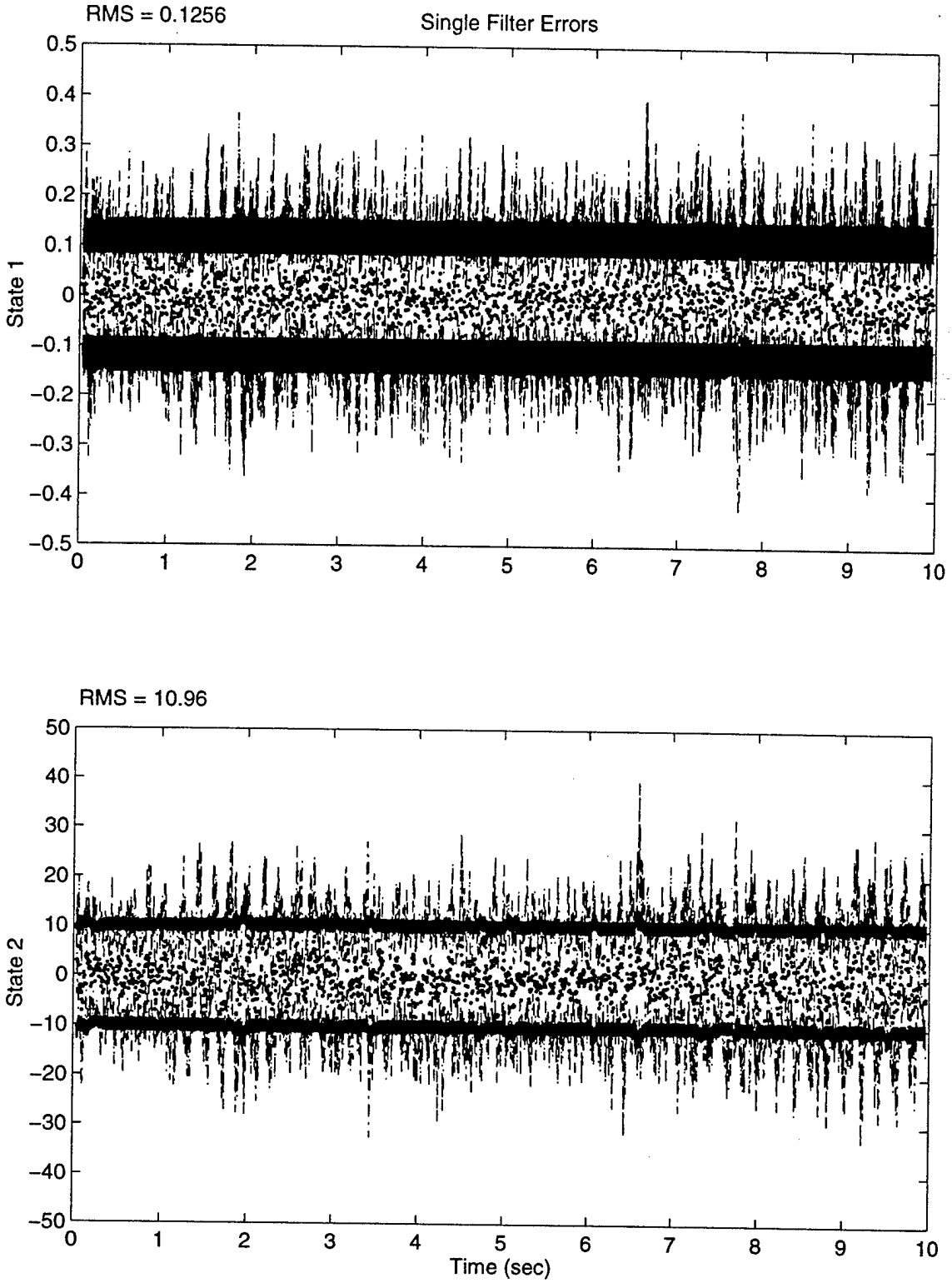


Figure 27. M^3AE Without IRDF State Estimation Performance: Test Case 1

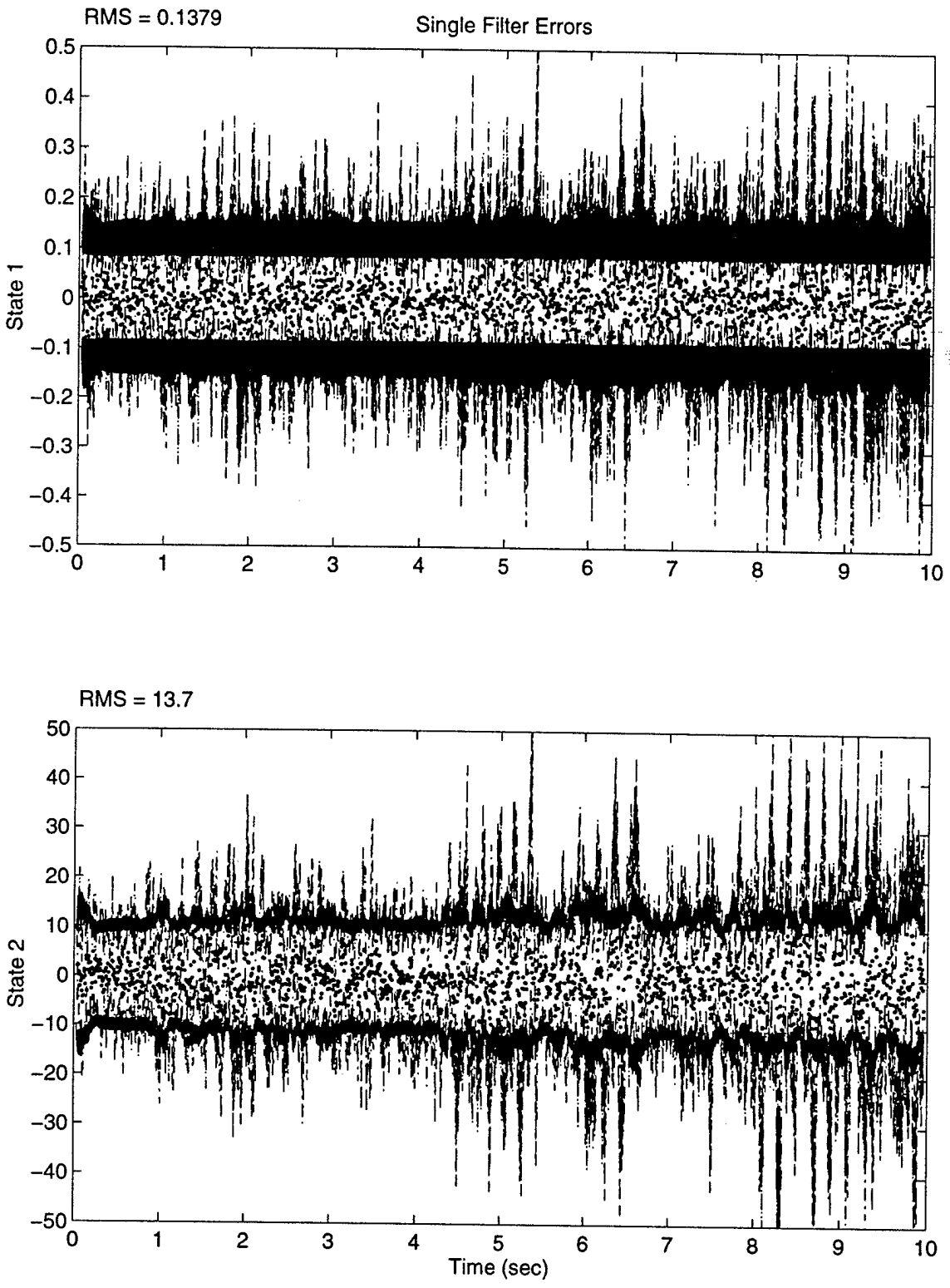


Figure 28. M^3AE With IRDF State Estimation Performance: Test Case 1

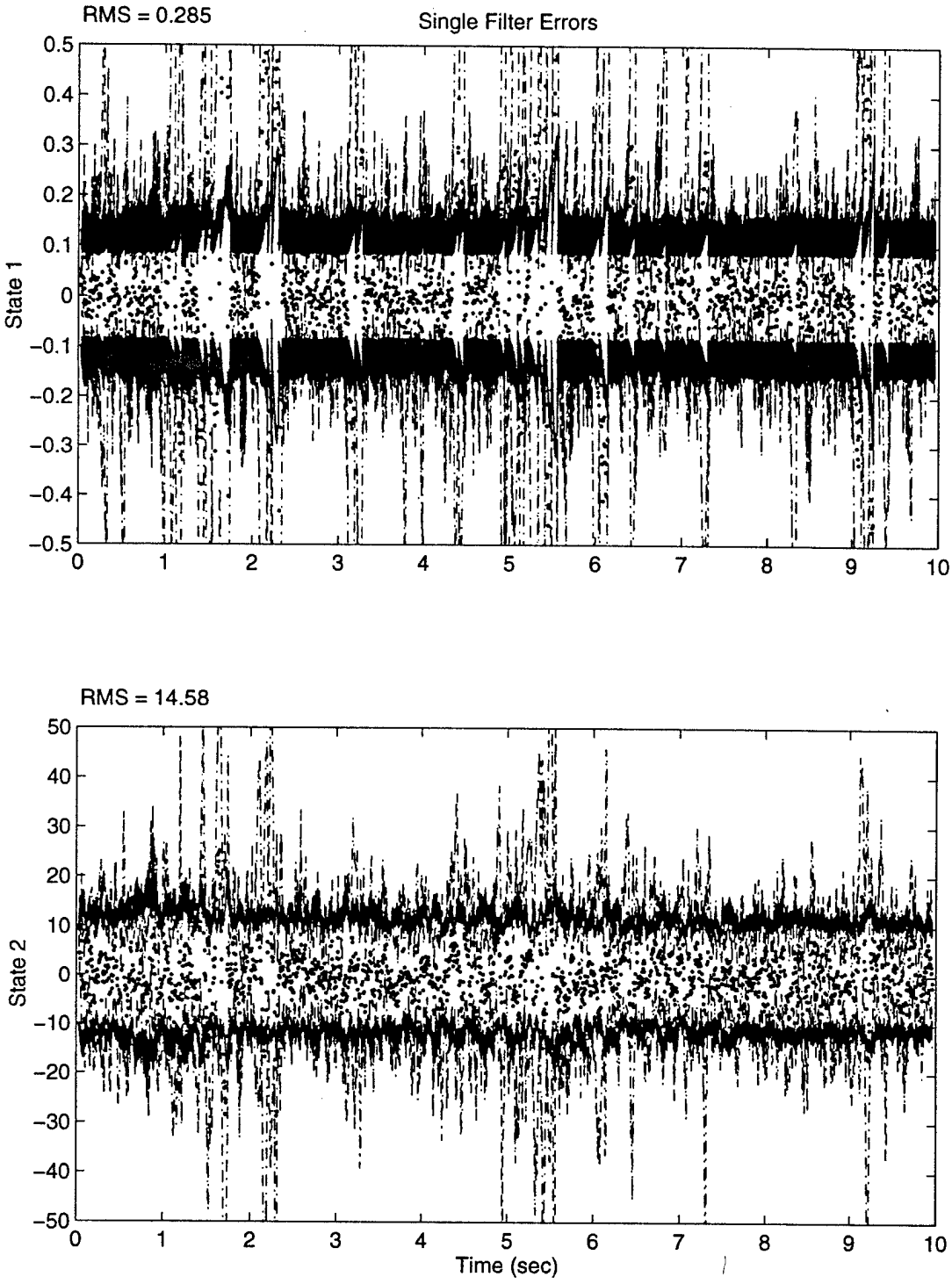


Figure 29. M^{3}AE With IRDF State Estimation Performance: Test Case 1, $\eta(t_i)$=0

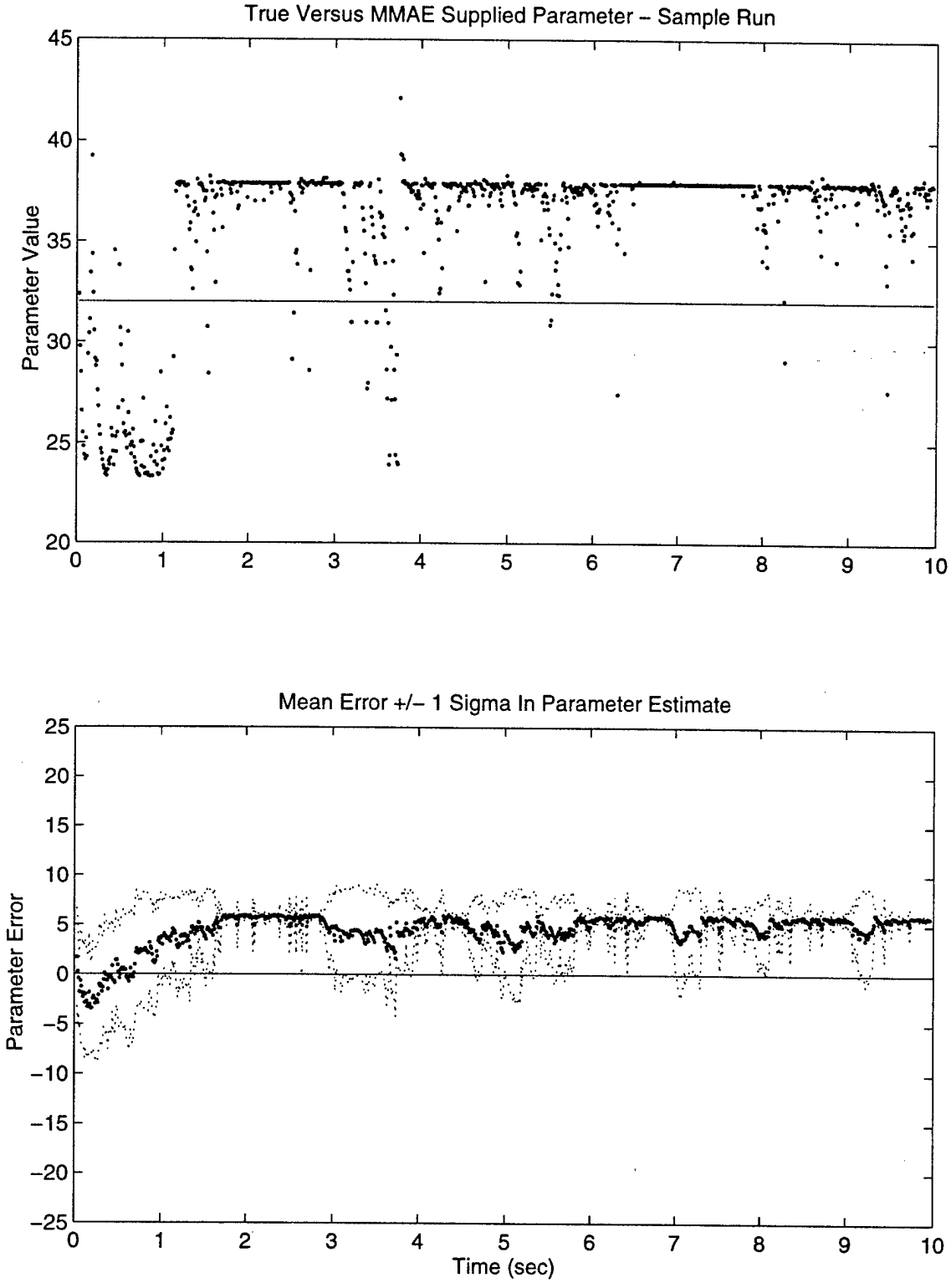


Figure 30. *MMAE/SDSEP* Parameter Estimation Performance: Test Case 1

the M^3AE's MMAE designed without IRDF, and in the first 4.2 seconds of the M^3AE's MMAE designed with IRDF. However, after about 4.2 seconds, the mean error in the M^3AE, designed with IRDF, tends towards zero (indicating better parameter estimation performance). This is a direct result of blending the *MMAE/SDPEP* 's a_2 and a_3 parameter values. Further notice, that this is the same point in the simulation where the state estimation performance from the blended MMAE estimate starts to deteriorate (see Figure 18 on page 108). This highlights the trade-off problem previously addressed in Chapters 1-3 concerning the difficulty involved in trying to provide accurate state and parameter estimates simultaneously. Additionally, the variance in the error grows larger after 4.2 seconds, but a zero-mean error with a large variance is preferred over a nonzero-mean error with a small variance, since in the real world the true parameter value is not known and any unknown bias may have a substantial negative impact on parameter estimation performance (and thus on the state estimation performance by the single Kalman filter within the M^3AE). This is evident upon inspection of the resulting M^3AE/IRDF's state estimation improvement, shown in Figure 28, over the *MMAE/SDPEP* /IRDF shown in Figure 18. Finally, despite setting $\eta(t_i) = 0$ throughout the simulation, the bottom plot in Figure 20 indicates that the parameter estimation performance of the *MMAE/SDPEP* /IRDF provides good parameter estimates. Thus, as discussed above and shown in Figure 29, the M^3AE produces good state estimation performance.

Summary. Overall, this test case clearly shows the benefits of the M^3AE architecture in estimating both parameters and states. The biggest benefit is seen in the M^3AE with IRDF, since this architecture produces a "near-zero-mean-error" blended parameter estimate, which in turn improves the state estimation performance. Additionally, the approximate M^3AE's covariance analysis shown in Figure 22 on page 117, indicates an upper bound on performance as compared to the actual performance shown in Figure 27 on page 128. This is a direct result of the bias in the *MMAE/SDPEP* without IRDF's parameter estimate shown in Figure 15 on page 105. Furthermore, the M^3AE clearly

outperforms the conventional *MMAE/SDSEP* and the M^{3}AE's *MMAE/SDSEP* in state and parameter estimation performance for this test case. Finally, the M^{3}AE designed without IRDF has slightly better state estimation performance than the M^{3}AE with IRDF, but the M^{3}AE implemented with IRDF provides the best overall parameter estimation performance for this test case due to the improved *MMAE/SDPEP* with IRDF's blending of the elemental filter parameter values.

4.1.3.2 Test Case 2: $a_T = 37.89$*, and Test Case 5:* $a_T = 28.40$

In these test cases the true parameter values, $a_T = 37.89$ and $a_T = 28.40$, exactly match the second elemental filters' parameter values, $a_2 = 37.89$ in the *MMAE/SDSEP* and $a_2 = 28.40$ in the *MMAE/SDPEP*. These cases provide the worst and best case scenarios for the M^{3}AE architecture. Performance is the worst when the true parameter value, $a_T = 37.89$, is *close* to the largest peak in the Sheldon optimization curve shown in Figure 14 on page 102, for optimized parameter estimation performance – thus poor parameter estimation performance is anticipated from the M^{3}AE for this particular value of a_T. Performance is the best when the true parameter value $a_T = 28.40$ exactly matches the second elemental filters' a_2. Again, an analysis of the actual state and parameter estimation performance is presented.

State Estimation Performance: Test Case 2. Figures 31 - 33 show the plots from each MMAE's blended state estimation performance. The temporally averaged RMS state estimation errors from each plot are summarized in Table 9 below.

Table 9. Test Case 2: MMAEs' Temporally Averged RMS State Estimation Errors

State	MMAE without IRDF	MMAE with IRDF	Conventional *MMAE/SDSEP*
x_1	0.8524	1.532	0.1775
x_2	36.05	60.42	12.51

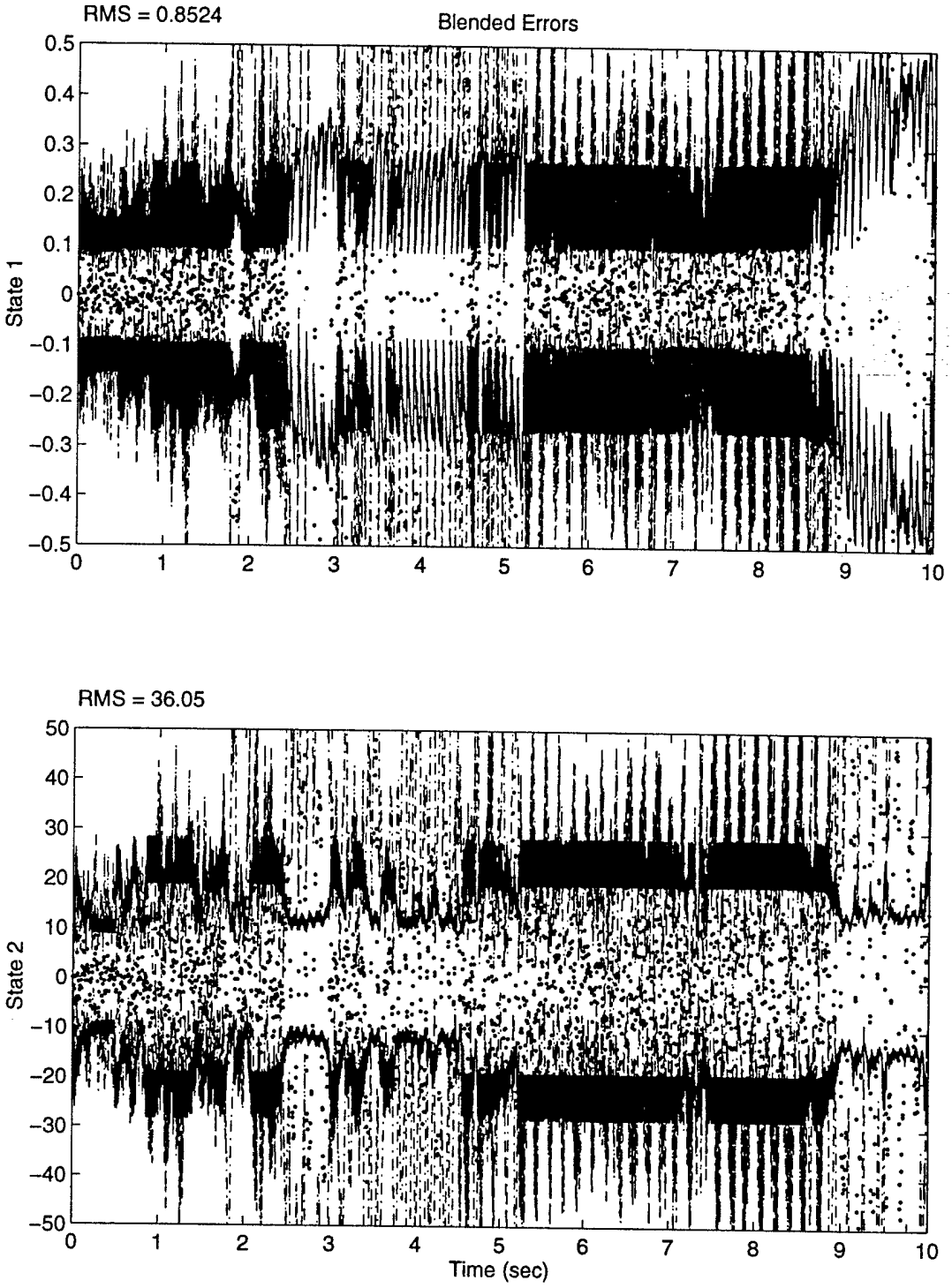


Figure 31. *MMAE/SDPEP* Without IRDF State Estimation Performance: Test Case 2

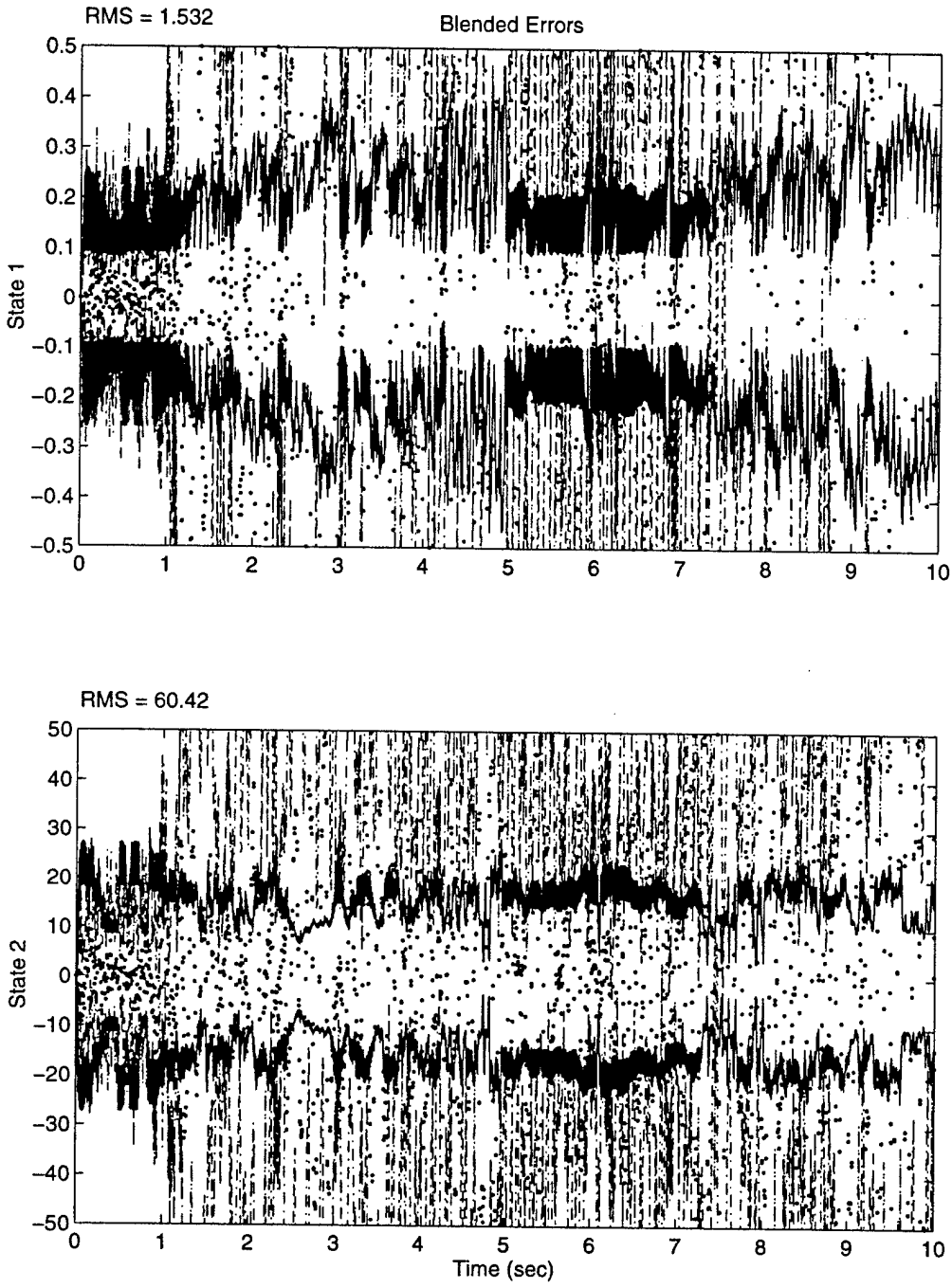


Figure 32. *MMAE/SDPEP* With IRDF State Estimation Performance: Test Case 2

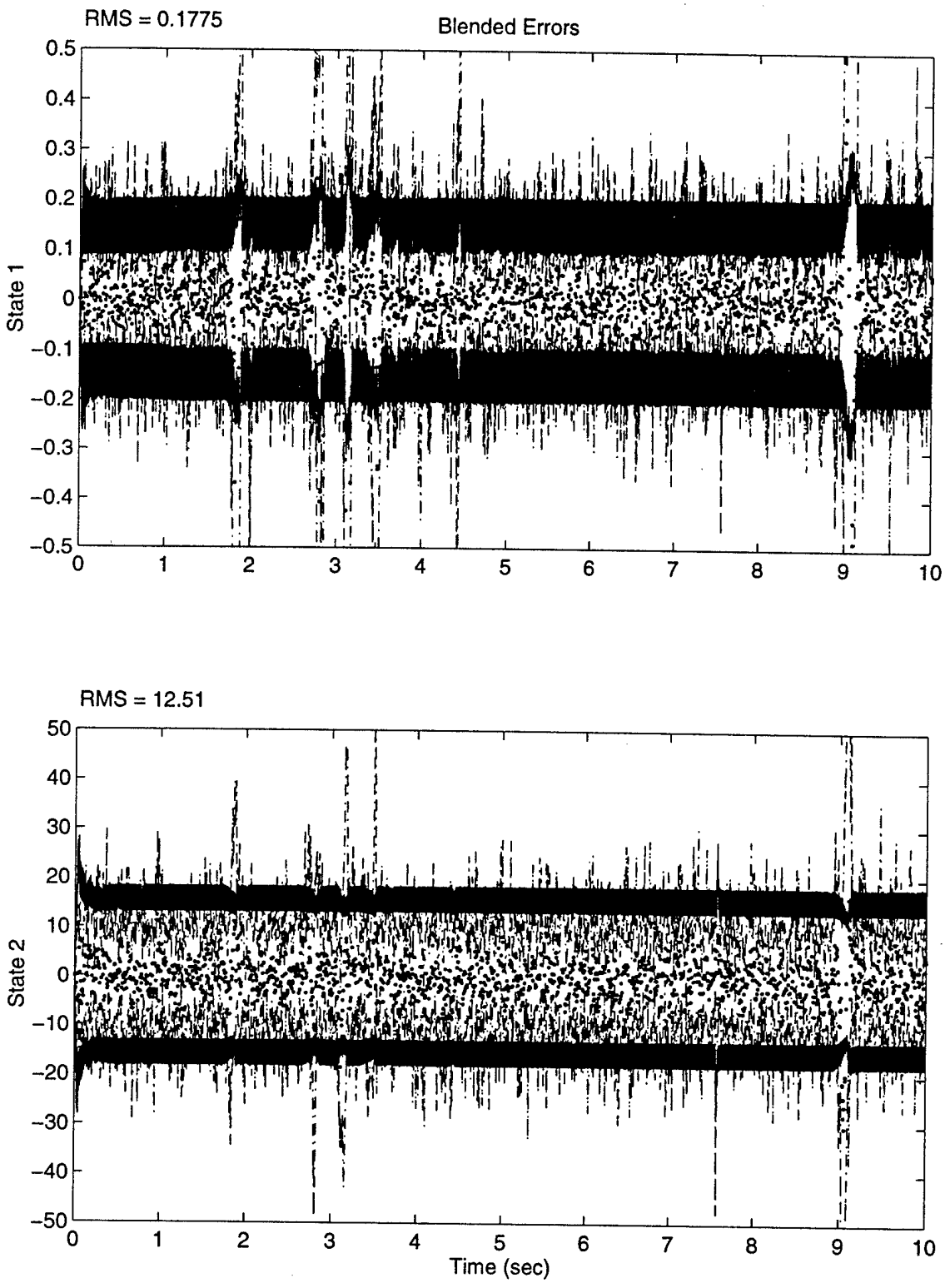


Figure 33. *MMAE/SDPEP* State Estimation Performance: Test Case 2

Clearly, the conventional *MMAE/SDSEP* outperforms the other two MMAEs. This is expected since the conventional MMAE's second elemental filter's parameter value $\mathbf{a}_2$ matches $\mathbf{a}_T$ exactly. As anticipated from knowledge of the Sheldon optimization curve shown in Figure 14 on page 102, both *MMAE/SDPEP*s suffer in this test case since $\mathbf{a}_T$ is located approximately halfway between the two *closest* elemental filters' $\mathbf{a}_j$'s in the bank. Thus, not only is the state estimation performance better for the *MMAE/SDSEP*, but the parameter estimation performance is also better (as shown in the *Parameter Estimation* subsection below). Additionally, note that the MMAE/IRDF's state estimation results are poor in this case, which is due to the decrease in the Kalman filter gains during the modulation of $\mathbf{Q}_d(t_i)$. Decreasing $\mathbf{Q}_d(t_i)$ tells the system to "trust" its internal dynamics model relatively more than the measurement updates entering the system, compared to the case without such decreased $\mathbf{Q}_d(t_i)$. Thus, for this test case, the filters' internal dynamics model of the system is degraded even further with the modulation of $\mathbf{Q}_d(t_i)$, causing even poorer state estimation performance, since even less weight is placed on the good measurements. This is shown clearly in Table 9 and Figures 31 and 32.

Next, note the M^{3}AE state estimation performance with or without IRDF, as indicated by the plots in Figures 34 and 35, and by the temporally averaged RMS state estimation errors from each plot shown in Table 10 below.

Table 10. Test Case 2: M^{3}AEs' Temporally Averged RMS State Estimation Errors

State	M^{3}AE without IRDF	M^{3}AE with IRDF
x_1	0.1836	0.3476
x_2	22.14	25.82

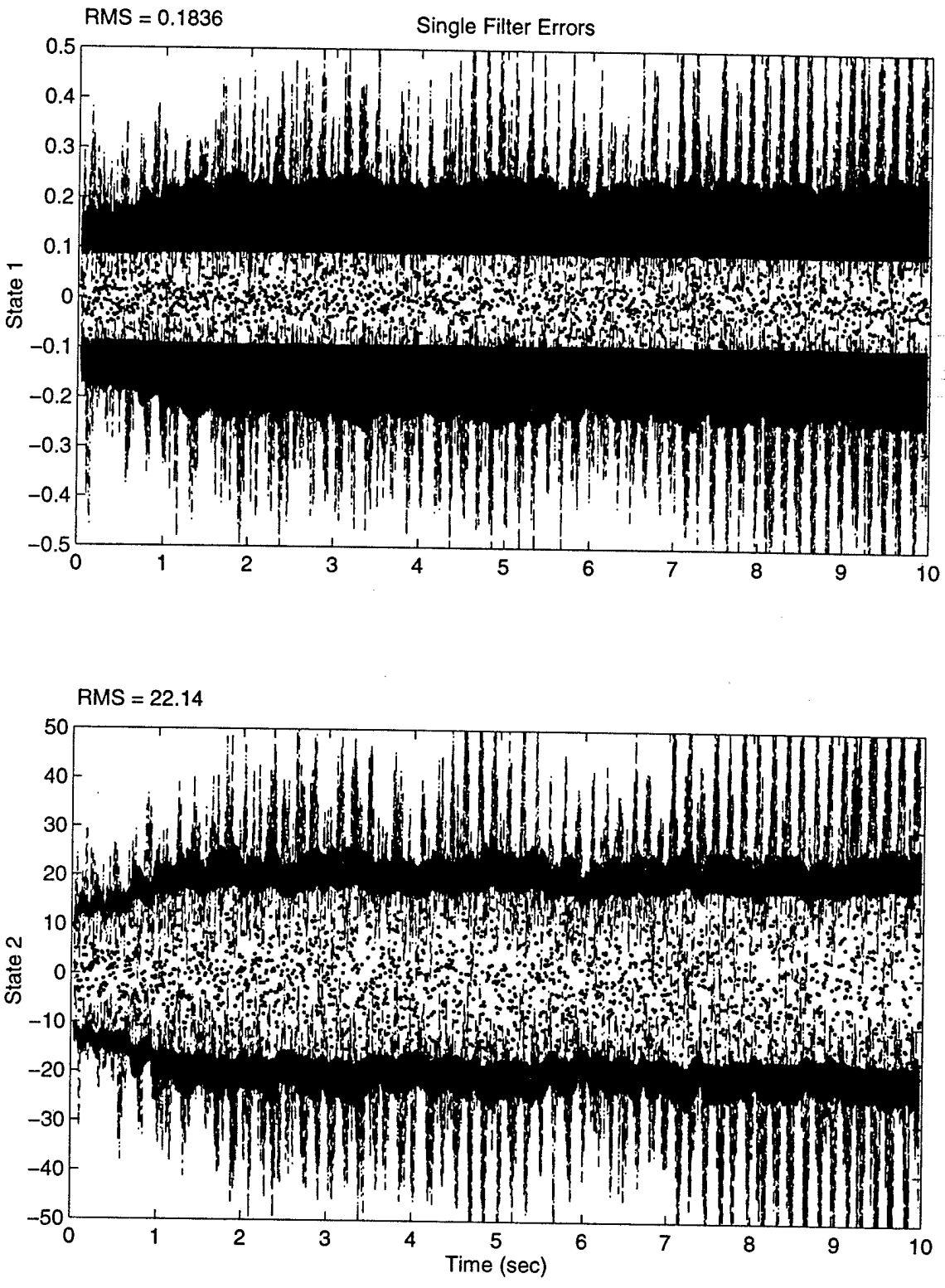


Figure 34. M^3AE Without IRDF State Estimation Performance: Test Case 2

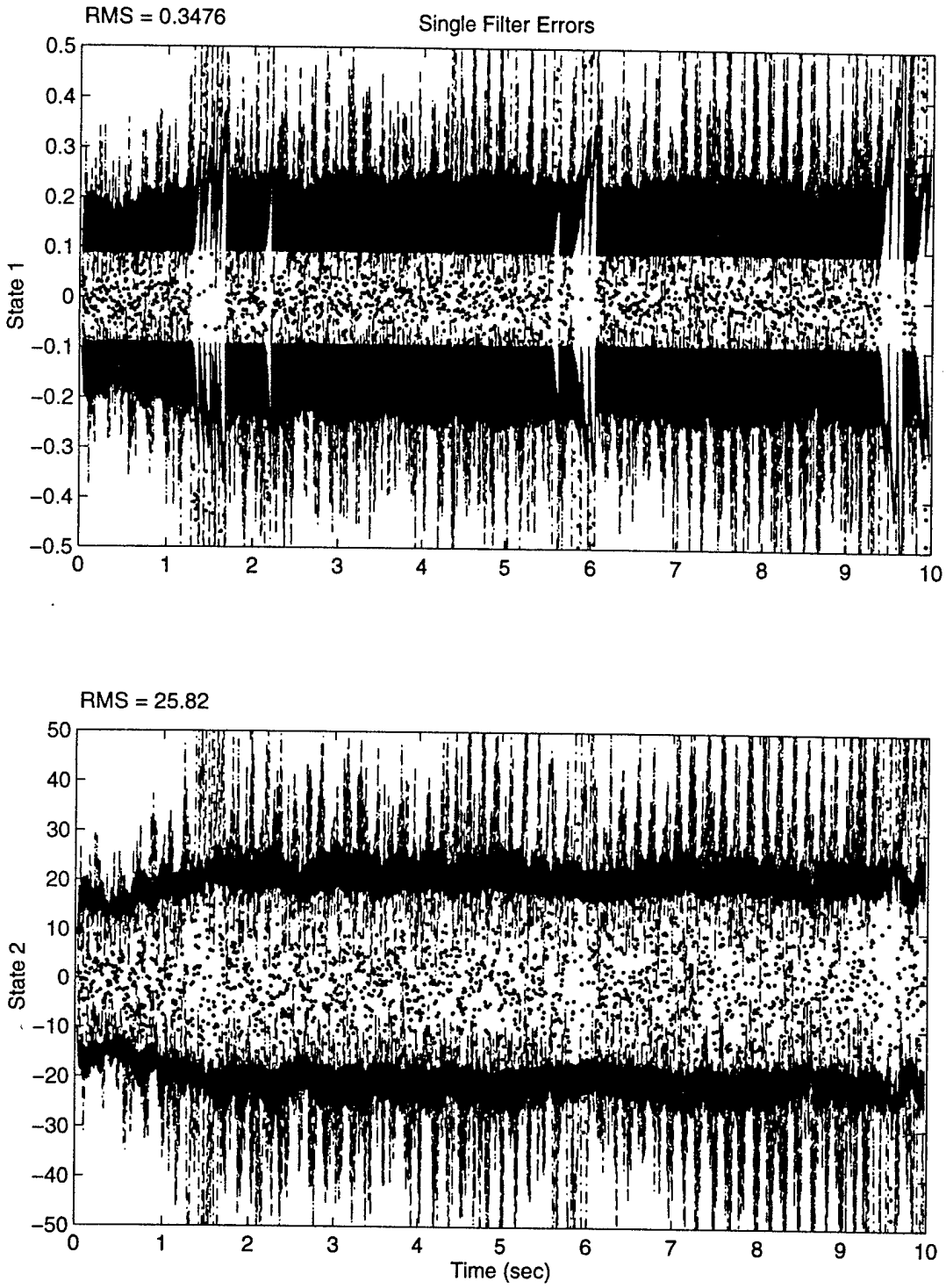


Figure 35. M^3AE With IRDF State Estimation Performance: Test Case 2

Clearly, the conventional *MMAE/SDSEP* performs better in state estimation performance than the M^{3}AE, as expected since poor parameter estimates are provided to the single Kalman filter in the M^{3}AE. However, notice that the state estimation performance improvement in using the M^{3}AE versus a stand-alone MMAE (with or without IRDF) "Sheldon-discretized" for parameter estimation performance, as seen in Table 9. Also note the similar performance between the two M^{3}AEs. Figures 34 and 35 indicate almost identical state estimation performance between both versions of the M^{3}AE. The exception occurs in state x_1 for the three regions (starting at approximately 1.3 seconds, 5.7 seconds, and 9.4 seconds) shown in Figure 35. In these regions, the mean errors and the associated error variances are larger than the corresponding errors in Figure 34 (without IRDF), which tends to drive the temporally averaged RMS state estimation errors upward, as shown in Table 10.

State Estimation Performance: Test Case 5. The temporally averaged RMS state estimation errors for the MMAEs and the M^{3}AEs for this test case are summarized in Table 11. In this case, both *MMAE/SDPEP* s clearly outperform the *MMAE/SDSEP*, as anticipated since $\mathbf{a}_T$ matches $\mathbf{a}_2$ exactly. The small differences between the *MMAE/SDPEP* with IRDF and without are due to the effects of the modulation of the dynamics driving noise $\mathbf{Q}_d(t_i)$ in the elemental filters. However, notice the M^{3}AEs' performance for both cases, which indicates the state estimation performance of the M^{3}AEs is virtually the same, as anticipated.

Table 11. Test Case 5: Temporally Averged RMS State Estimation Errors

State	MMAE without IRDF	MMAE with IRDF	Conventional *MMAE/SDSEP*		M^{3}AE without IRDF	M^{3}AE with IRDF
x_1	0.1140	0.1409	0.2139		0.1051	0.1065
x_2	7.630	8.496	12.29		7.519	7.693

Also note, that the only figure shown for this test case is a single plot of the M^{3}AE's state estimation performance as shown in Figure 36, since no additional insight is gained by displaying the other figures associated with this test case. This figure is presented in order to compare the actual performance of the M^{3}AE to that predicted by the M^{3}AE's approximate covariance analysis tool shown in Figure 24 on page 119. These figures highlight the improvement in the prediction capability of the M^{3}AE's approximate covariance analysis tool when the bias in the parameter estimate is small.

Parameter Estimation Performance: Test Case 2. Figures 37 - 39 show the plots of each MMAE's blended parameter estimation performance. Again notice that, in each case, the top plots in these figures indicate that the MMAE-supplied parameter estimate converges to the $\mathbf{a}_j$ *closest* to $\mathbf{a}_T$ in the "Baram distance measure sense," as anticipated (for both *MMAE/SDPEP* s, the *closest* elemental filter is $\mathbf{a}_3 = 49.9$, whereas the *MMAE/SDSEP* 's *closest* elemental filter is $\mathbf{a}_2 = 37.89$). Notice, however, that the *MMAE/SDPEP* 's parameter estimate, $\hat{\mathbf{a}}$, is slightly lower than the 49.9 value associated with $\mathbf{a}_3$, while the *MMAE/SDSEP* 's parameter estimate, $\hat{\mathbf{a}}$, is slightly higher than the 37.89 value associated with $\mathbf{a}_2$; this is anticipated and is a direct result of placing a minimum probability on each of the MMAEs' elemental filters. Furthermore, notice the mean errors in the parameter estimates shown in the bottom plot in each figure. There are large biases in the parameter estimates from both *MMAE/SDPEP* s, as expected, given the value of $\mathbf{a}_T$ relative to the *MMAE/SDPEP* elemental filters' assumed $\mathbf{a}_j$'s.

Parameter Estimation Performance: Test Case 5. As anticipated, the parameter estimation is much better in both *MMAE/SDPEP* s as compared to the *MMAE/SDSEP.* Just the opposite occurs in this test case as compared to test case 2. In this test case, the *MMAE/SDSEP* experiences the bias while both *MMAE/SDPEP* s have zero-mean-error estimates.

Summary. Clearly, the conventional *MMAE/SDSEP* outperforms both M^{3}AEs in this test case 2, as was anticipated. However, this test case again highlights the performance improvement in state

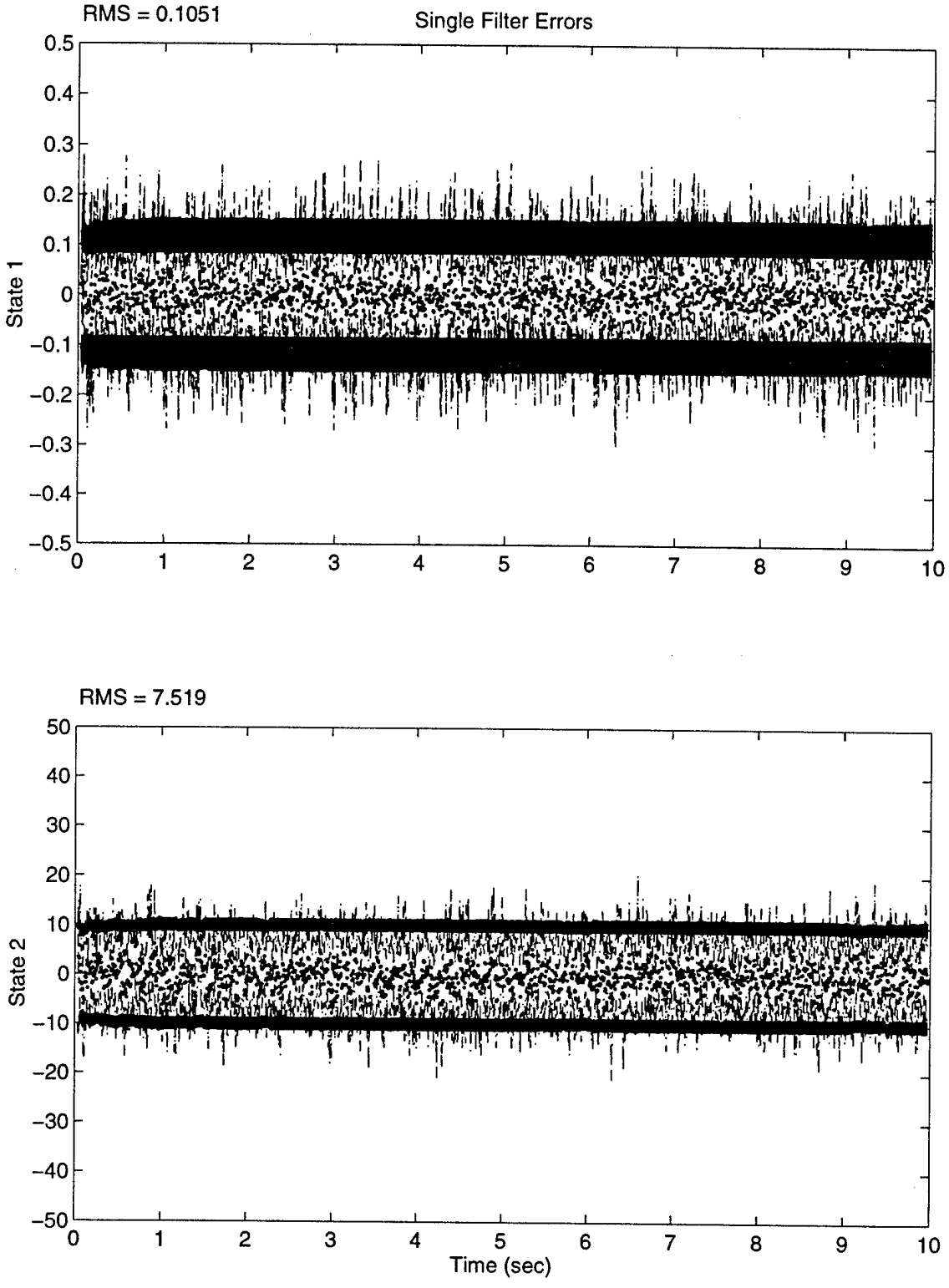


Figure 36. M^{3}AE Without IRDF State Estimation Performance: Test Case 5

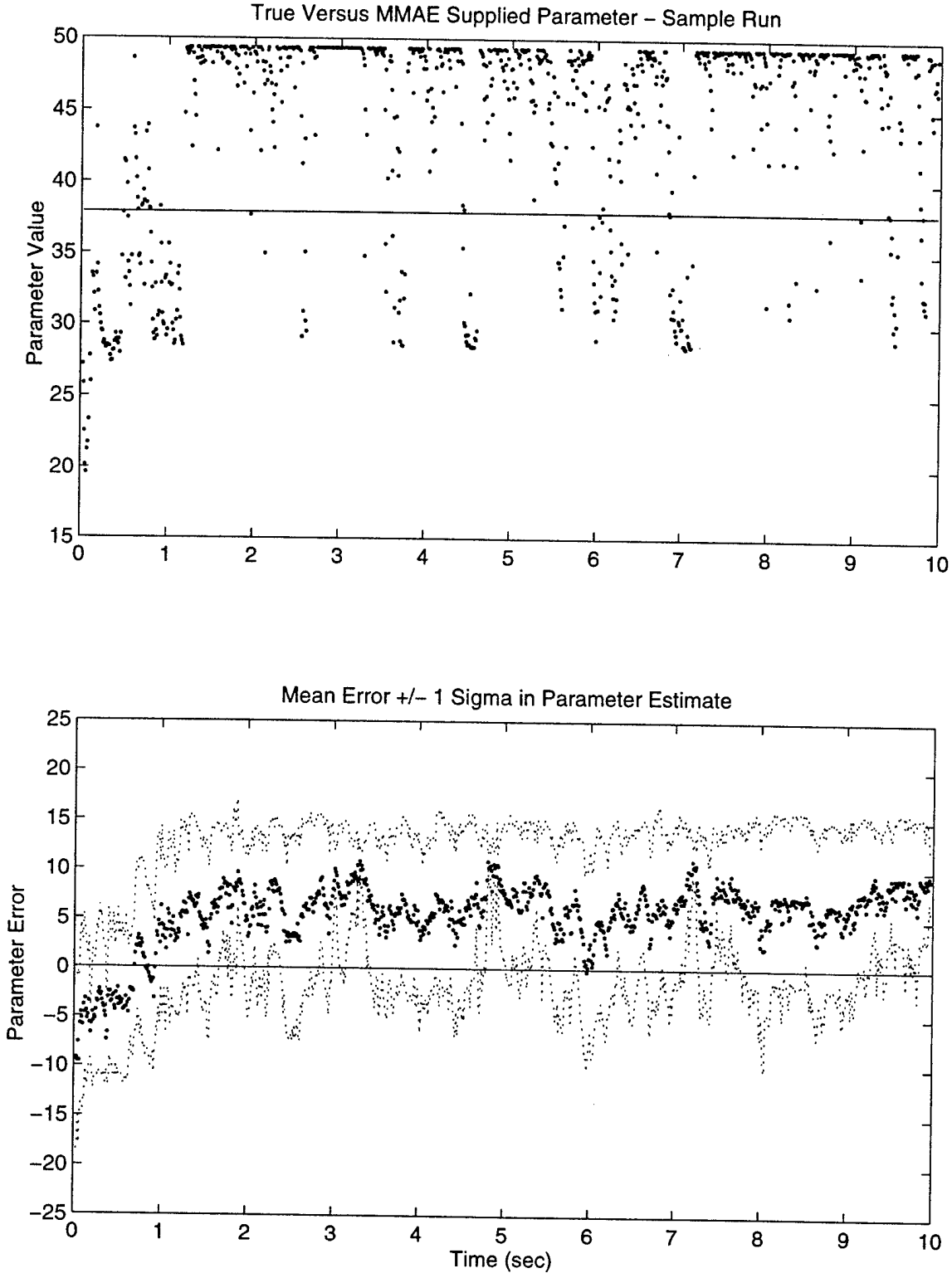


Figure 37. *MMAE/SDPEP* Without IRDF Parameter Estimation Performance: Test Case 2

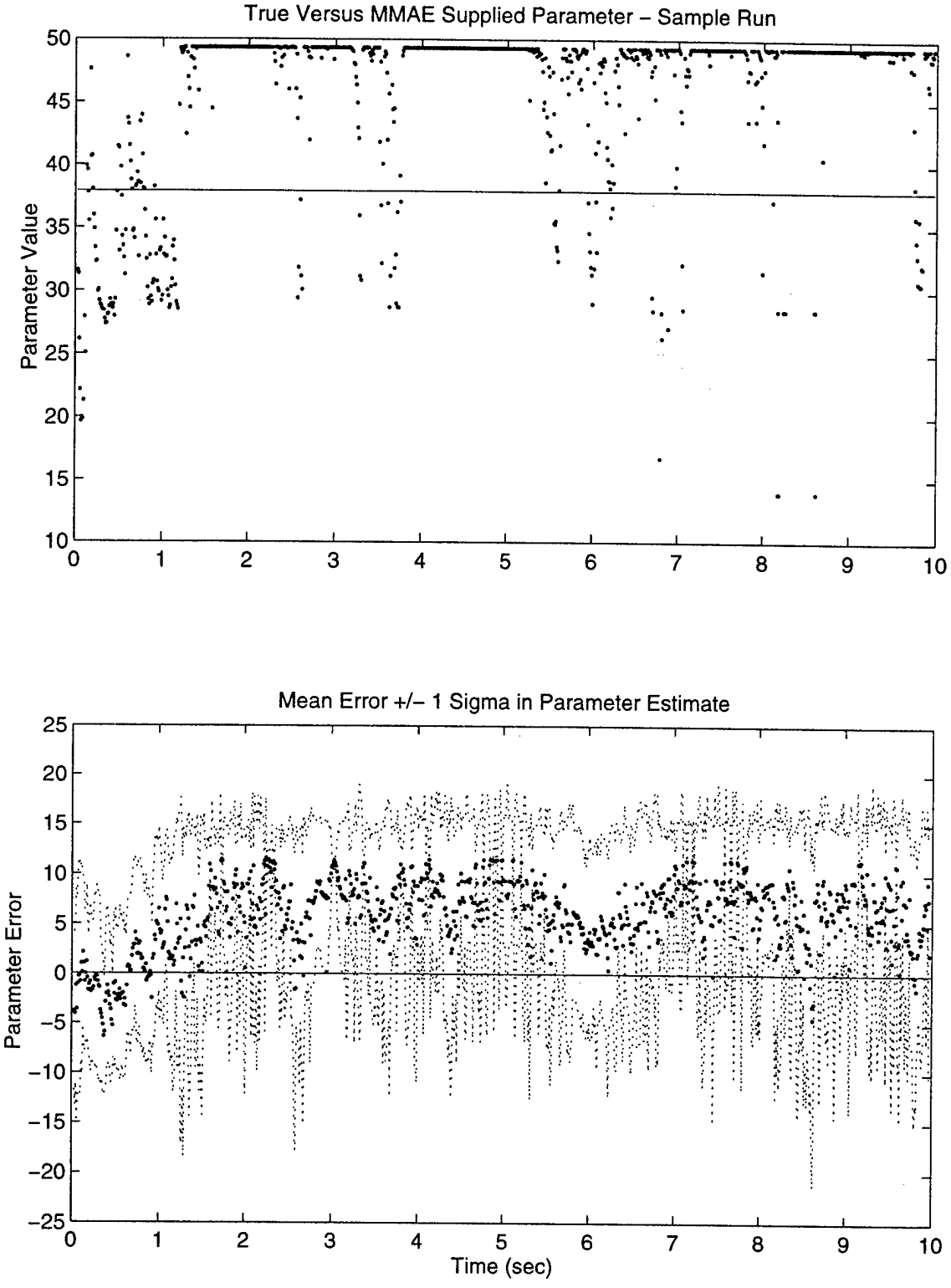


Figure 38. *MMAE/SDPEP* With IRDF Parameter Estimation Performance: Test Case 2

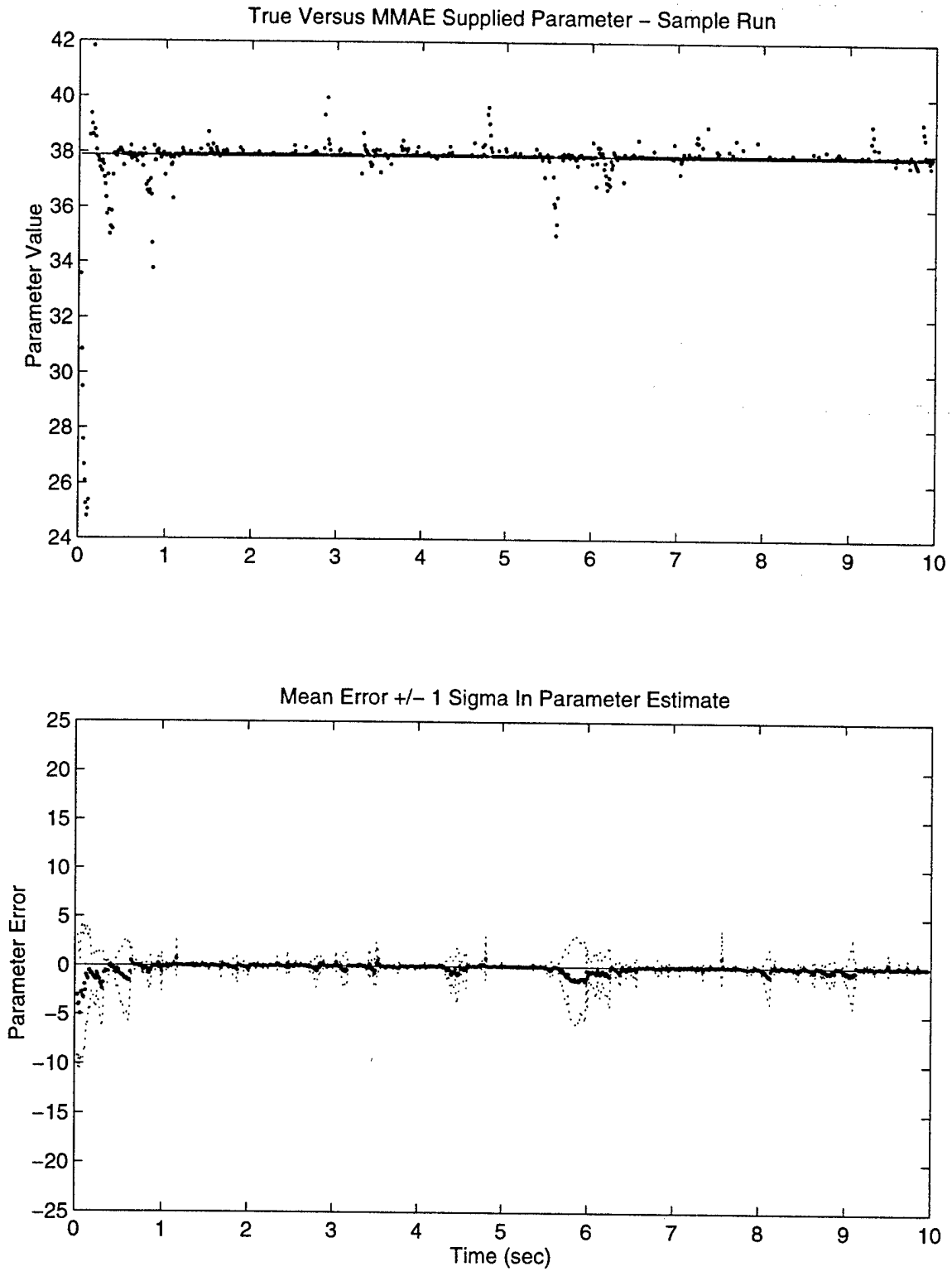


Figure 39. *MMAE/SDSEP* Parameter Estimation Performance: Test Case 2

estimation as a result of using the M^{3}AE architecture – the M^{3}AE state estimation performance is significantly better than that of either of the M^{3}AEs' MMAEs "Sheldon-discretized" for parameter estimation.

Additionally, test case 2 uncovers an important concern in the initial *MMAE/SDPEP* design. If it is anticipated that $\mathbf{a}_T$ will be near a peak in the Sheldon optimization curve the majority of the time, then the optimization should be reaccomplished, after "fixing" one of the parameter values to the anticipated $\mathbf{a}_T$. This should in turn improve the parameter estimation performance and the subsequent state estimation performance in the M^{3}AE.

Furthermore, note that the approximate M^{3}AE's covariance analysis shown in Figure 23 on page 118 is "biased high" compared to the actual performance shown in Figure 34. Therefore, this case could warrant using the M^{3}AE approximate covariance analysis tool with Section 3.3.1.3's bias term evaluation as an approximation to the *total* $\mathbf{\Psi}$ (since now $\mathbf{m}\mathbf{m}^T$ would dominate $\mathbf{P}$ in $\mathbf{\Psi} = \mathbf{P} + \mathbf{m}\mathbf{m}^T$) to evaluate the achievable M^{3}AE state estimation accuracy for such a worst case $\mathbf{a}_T$ *far* from any $\mathbf{a}_j$'s in the current discretization.

Finally, test case 5 provides results which are just the opposite of those seen in test case 2, which was anticipated given the *location* of $\mathbf{a}_T$ in this test case. This further emphasizes the importance in the *placement* of discrete parameters in the parameter space for defining the elemental filters in the bank. Furthermore, this concern, plus other similar concerns, will support a desire to consider moving-bank versus fixed-bank M^{3}AE algorithms, as will be discussed further in Chapter 5.

4.1.3.3 Test Case 8: $\mathbf{a}_T = 35.0$ *and* 30.385

In this test case the true parameter value, $\mathbf{a}_T$, undergoes a step change from 35.0 to 30.385, three seconds into the simulation. These values represent a small change in parameter value ω_n which may occur due to a minor perturbation affecting the system. The purpose of this test case

is to investigate the M^{3}AE's ability to handle parameter changes. Again, an analysis of the actual state and parameter estimation performance is conducted, with an emphasis placed on performance during the step change in the parameter value.

State Estimation Performance. Figures 40 - 42 show the plots from each MMAE's blended state estimation performance. The temporally averaged RMS state estimation errors from each plot are summarized in Table 12 below.

Table 12. Test Case 8: MMAEs'Temporally Averged RMS State Estimation Errors

State	MMAE without IRDF	MMAE with IRDF	Conventional *MMAE/SDSEP*
x_1	0.1231	0.5701	0.2482
x_2	10.86	27.79	14.65

The *MMAE/SDSEP* provides good state estimation performance, but it is difficult to determine when the step change occurs. However, upon closer inspection of Figure 42, an increase in the "spread" of the mean and the mean $\pm$ one sigma values starts at $t_i = 3$. This occurs since the *MMAE/SDSEP* does not compensate for the decrease in the true parameter value adequately. The *MMAE/SDSEP* 's second elemental filter ($\mathbf{a}_2 = 37.89$) receives the majority of the probability weighting throughout the simulation, even though the true parameter value decreases – this is evident upon inspection of the top plot in Figure 47 on page 157, where the parameter estimate $\hat{\mathbf{a}}_{MMAE}$ remains equal to $\mathbf{a}_2$ throughout the majority of the simulation.

The M^{3}AE's MMAE, implemented without IRDF, performs the best in state estimation and also adapts to the step change in the true parameter value $\mathbf{a}_T$ which occurs at $t_i = 3$ seconds into the simulation. State estimation performance also improves, which is expected since the new value of $\mathbf{a}_T = 30.385$ is very *close* to $\mathbf{a}_2 = 28.40$ (consider Figure 45 on page 155). For this specific

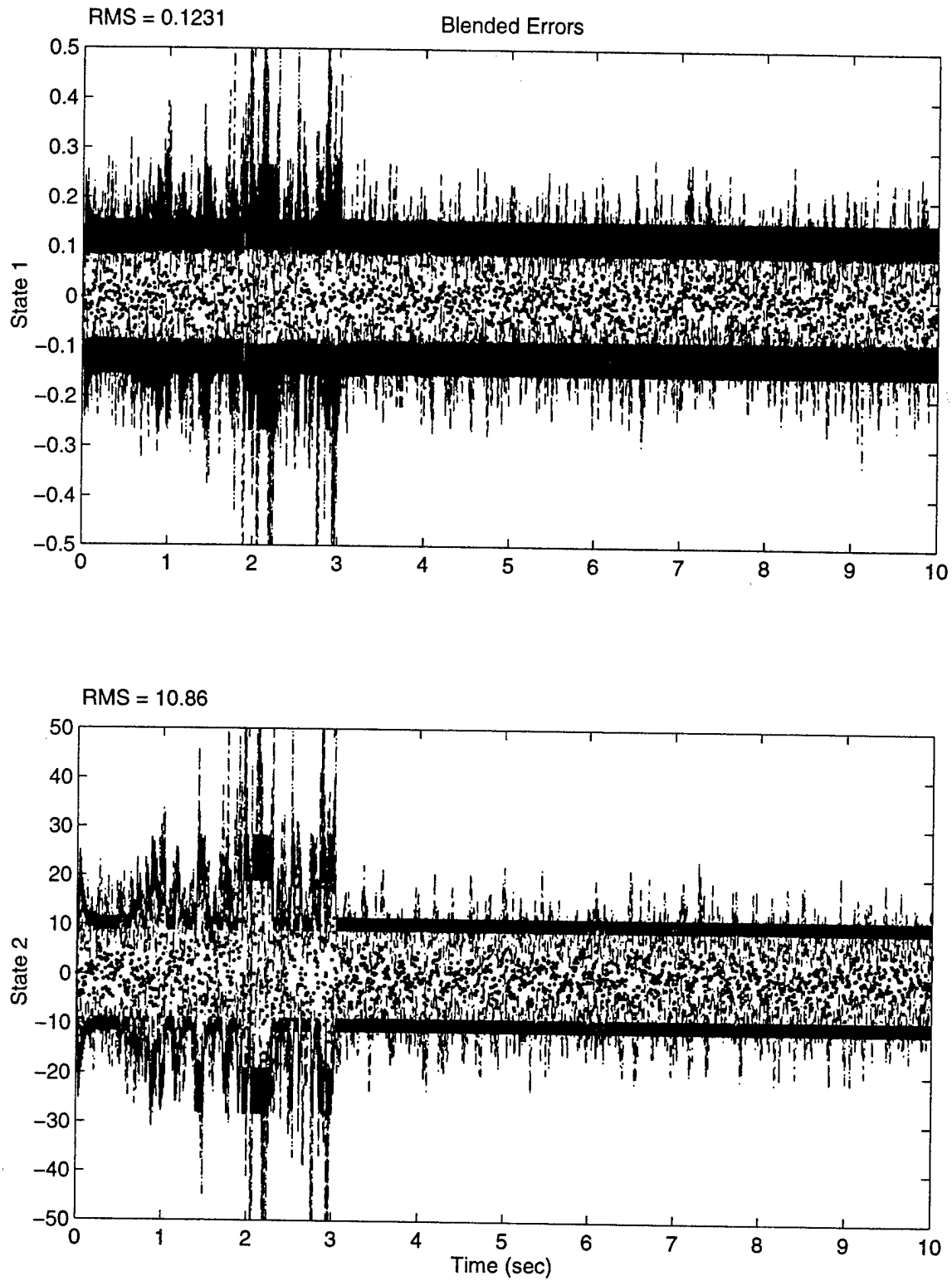


Figure 40. *MMAE/SDPEP* Without IRDF State Estimation Performance: Test Case 8

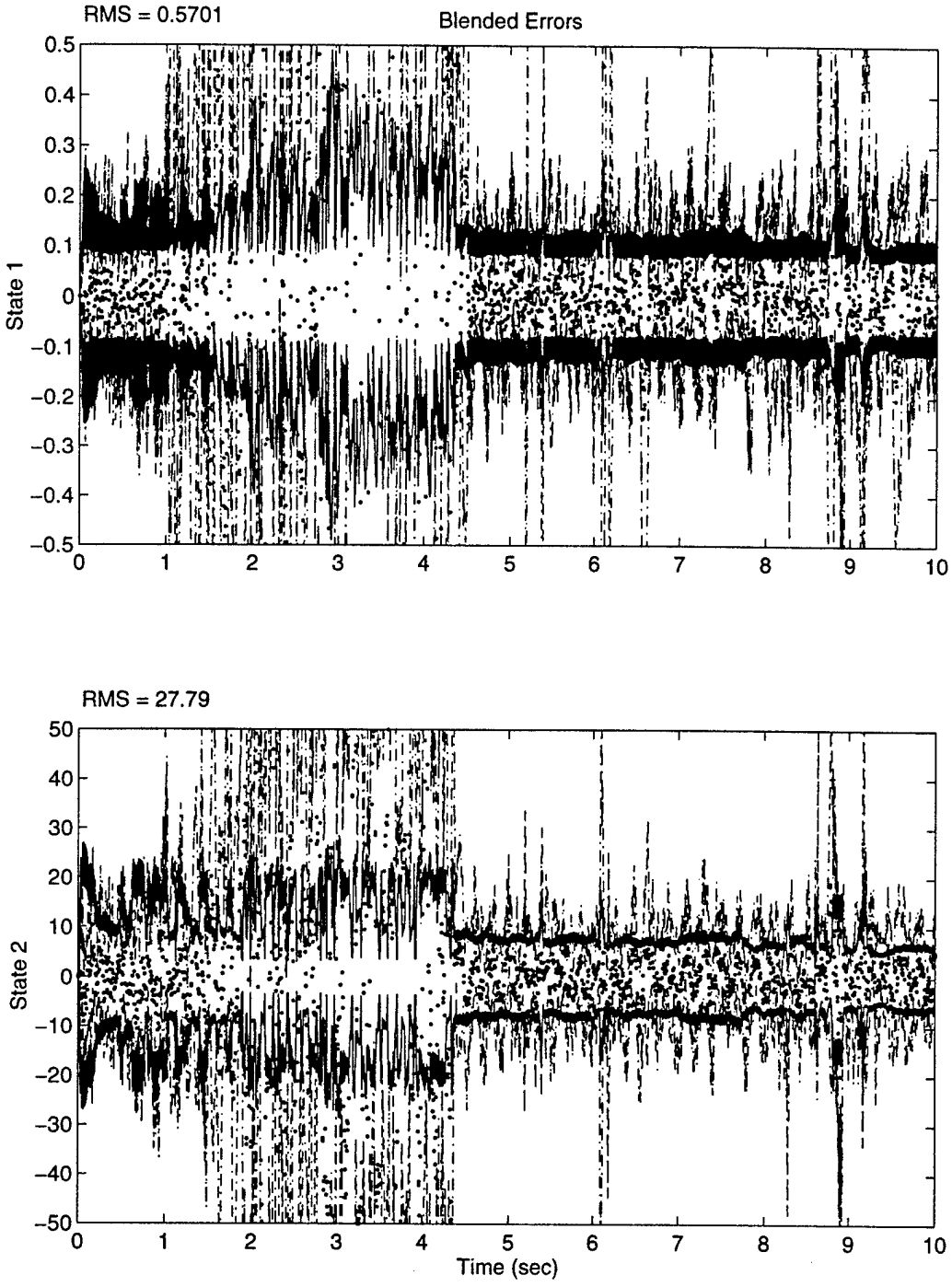


Figure 41. *MMAE/SDPEP* With IRDF State Estimation Performance: Test Case 8

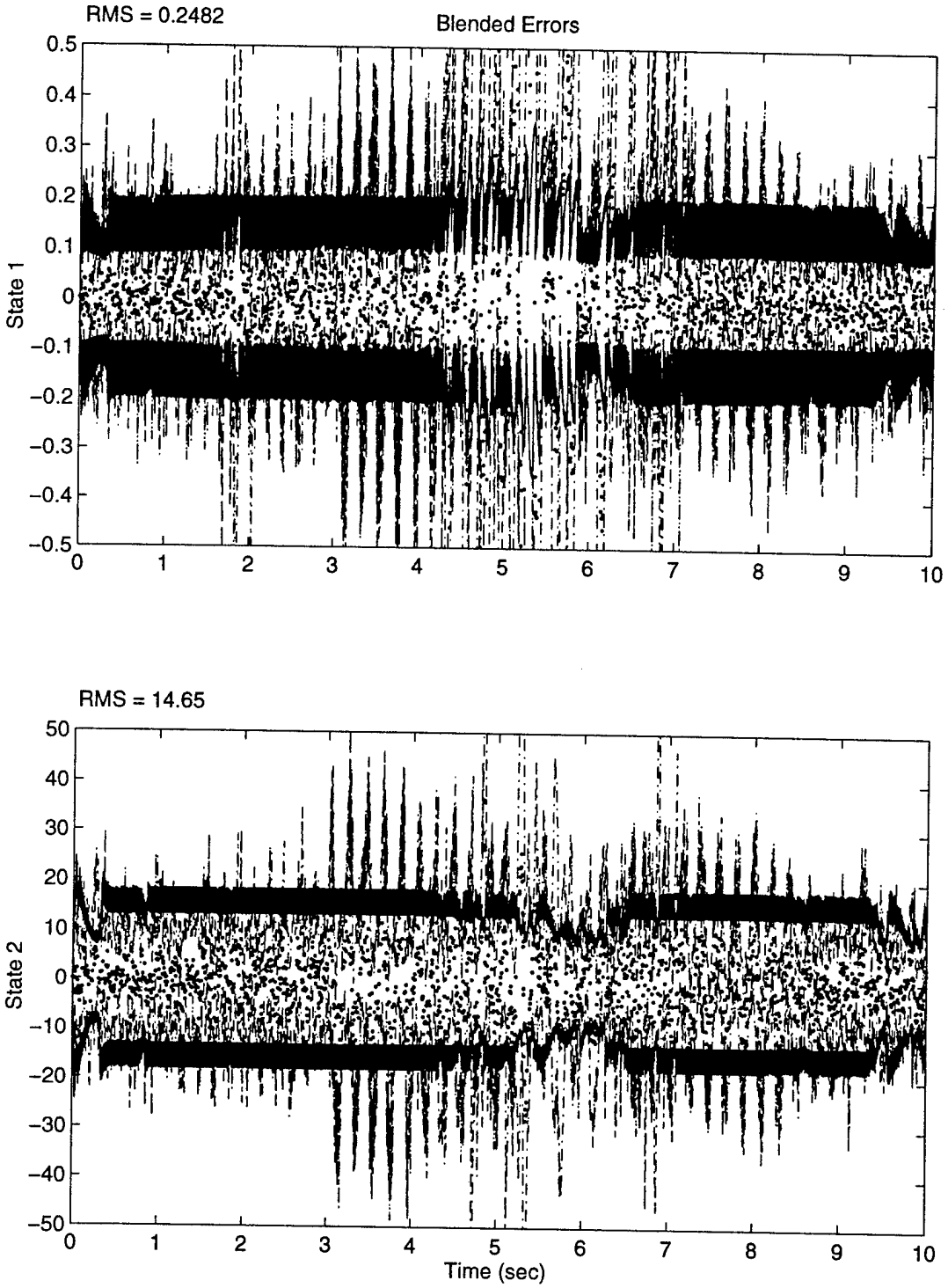


Figure 42. *MMAE/SDSEP* State Estimation Performance: Test Case 8

case, the step change of a_T at $t_i = 3$ seconds, *also* yields a new a_T value *further* from any a_j in the *MMAE/SDSEP* (see Figure 47 on page 157), so its state estimation performance degrades at the point where the *MMAE/SDPEP* s and the M^{3}AE's state estimation performances improve. Such is clearly not always the case.

Finally, the M^{3}AE's MMAE, implemented with IRDF, performs the worst in state estimation performance, as expected. In addition, there is almost a 1.3 second delay before the step change becomes obvious, as shown in Figure 41. The delay is a direct result of using IRDF to demodulate the filter gains, which tends to decrease the state estimation performance, as was shown in the previous test cases.

Next, note the M^{3}AE state estimation performance, with or without IRDF, as indicated by the plots in Figures 43 and 44, and by the temporally averaged RMS state estimation errors from each plot shown in Table 13 below.

Table 13. Test Case 8: M^{3}AEs' Temporally Averged RMS State Estimation Errors

State	M^{3}AE without IRDF	M^{3}AE with IRDF
x_1	0.1222	0.1374
x_2	10.42	12.59

Again, both M^{3}AEs outperform the *MMAE/SDSEP*, and *both* indicate the onset of the step change in a_T as shown in Figures 43 and 44. However, the M^{3}AE implemented without IRDF, provides a much more obvious indication of the step change than the M^{3}AE implemented with IRDF. Nevertheless, the fact that the M^{3}AE, implemented with IRDF, even indicates the step change is significant since the M^{3}AE's MMAE, implemented with IRDF, did not indicate the immediate onset of the step change in a_T. This further highlights the state estimation performance benefit associated with

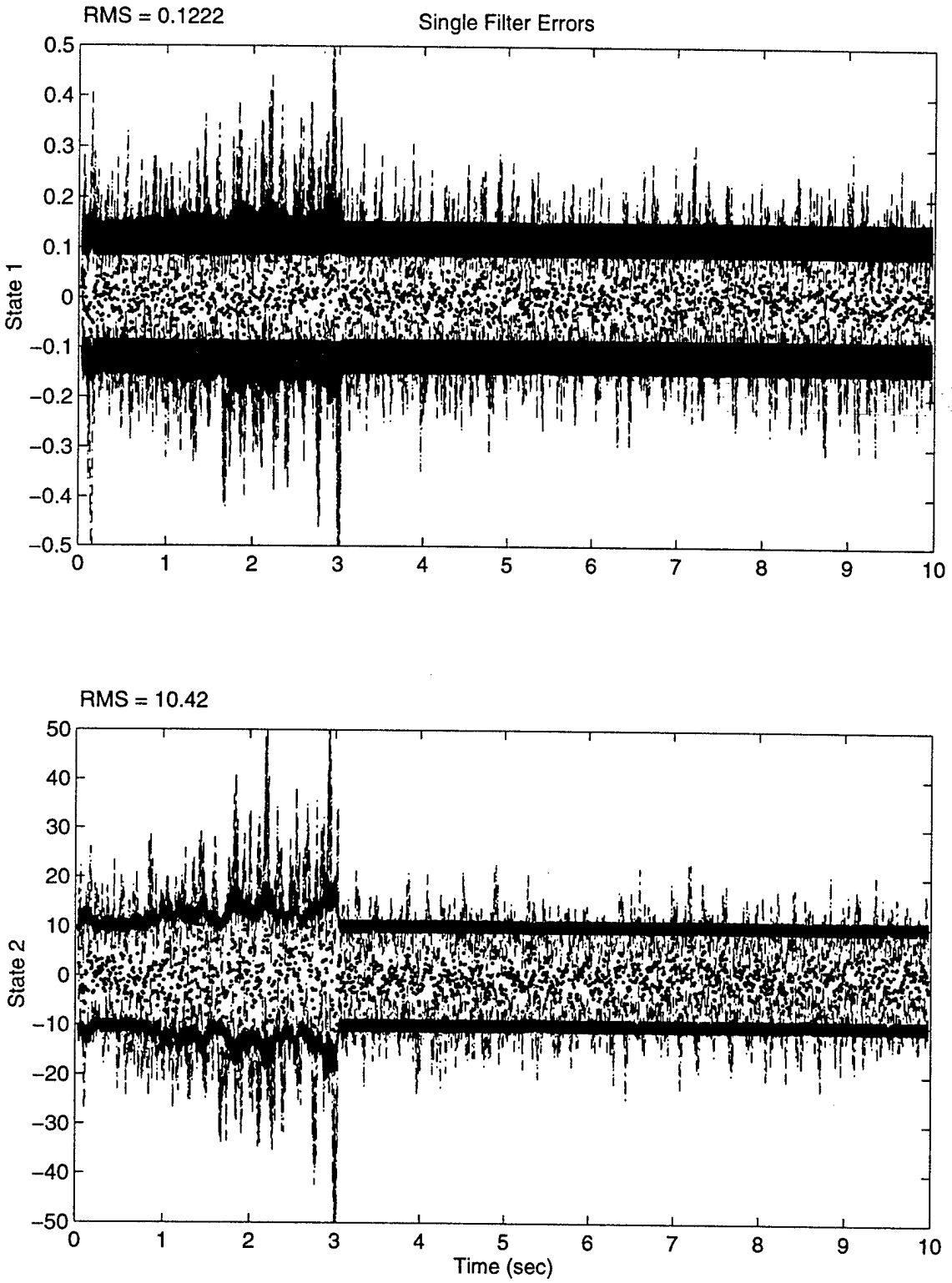


Figure 43. M^3AE Without IRDF State Estimation Performance: Test Case 8

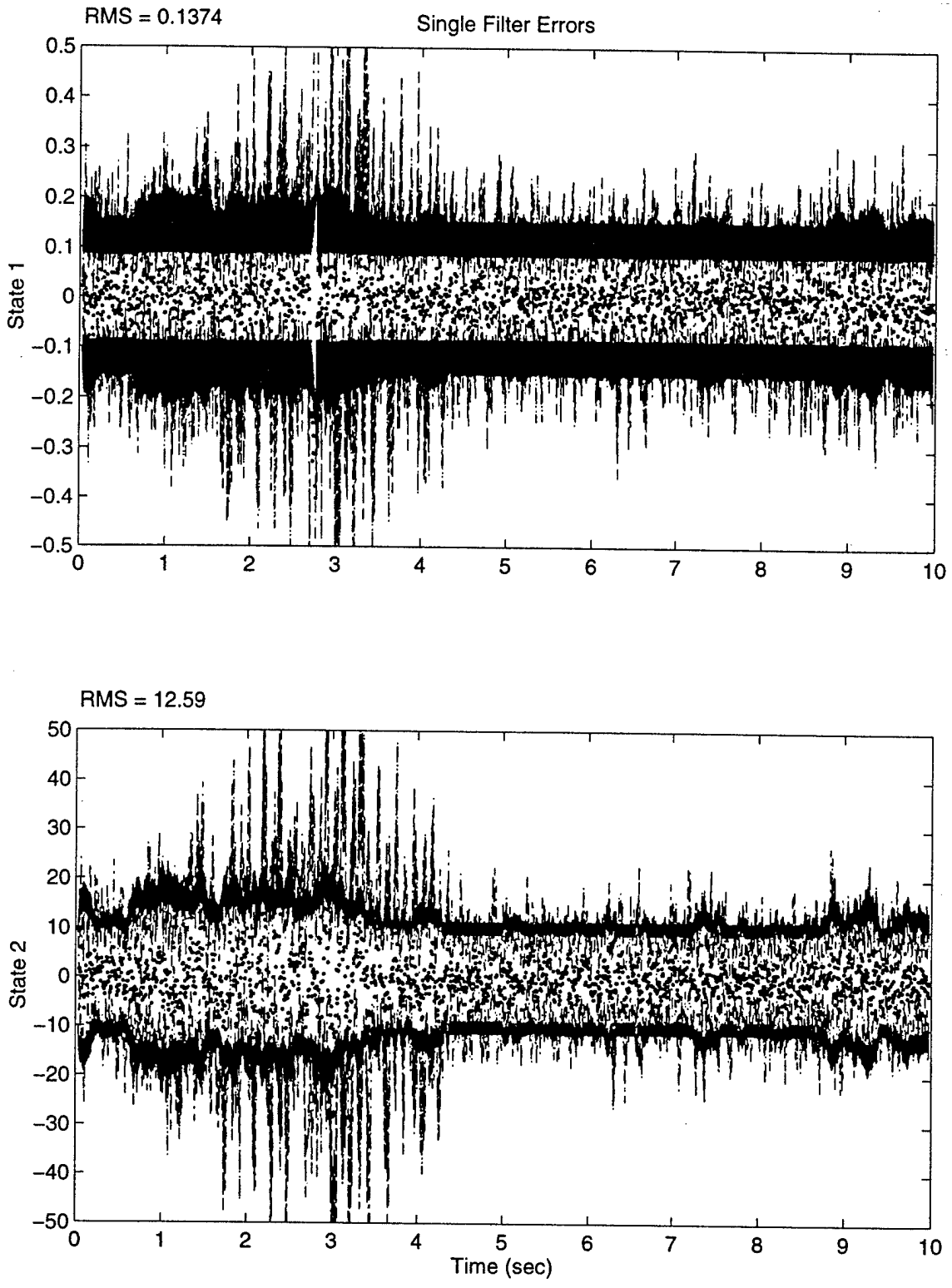


Figure 44. M^3AE With IRDF State Estimation Performance: Test Case 8

providing a good parameter estimate $\hat{\mathbf{a}}_{MMAE}$ (as discussed below) to the M^{3}AE's single Kalman filter tuned for accurate state estimation performance.

Parameter Estimation Performance. Figures 45 - 47 show the plots of each MMAE's blended parameter estimation performance. As in the previous test cases, notice that the top plots in these figures indicate that the MMAE-supplied parameter estimate converges to the $\mathbf{a}_j$ *closest* to $\mathbf{a}_T$ in the "Baram distance measure sense," as anticipated (for the *MMAE/SDPEP* s, $\mathbf{a}_2 = 28.40$, while for the *MMAE/SDSEP*, $\mathbf{a}_2 = 37.89$). Notice the mean errors in the parameter estimates shown in the bottom plot in each figure. There exist distinct biases of different magnitude for all time in the parameter estimates from all three MMAEs. Thus, all three filters look somewhat oblivious to the parameter change. However, some interesting conclusions can be drawn from this example. For instance, in looking at Figure 45, notice that the parameter estimation "cleans up" at $t = 3$ seconds due to the step change. But this is coincidental and it would not really be fair to claim improvement, since if the step change had been in the reverse direction (i.e., from 30.385 to 35), the state and parameter estimation performance would have degraded in both of the *MMAE/SDPEP* s, whereas the state and parameter estimation performance would have improved in the *MMAE/SDSEP*. This emphasizes the discussion presented earlier concerning the proximity of the MMAE's *closest* $\mathbf{a}_j$ to $\mathbf{a}_T$, and will motivate the desire (discussed more fully in Chapter 5) to consider moving-bank versus fixed-bank M^{3}AE algorithms in practice. Also notice that, as discussed in test case 1, those areas in the simulation where the state estimation performance of the M^{3}AE's MMAE, designed with IRDF, is poor (see Figure 41, from 3 to 4.3 seconds), the parameter estimation error performance is good (see Figure 46, especially from 3.4 to 4.3 seconds, where the mean errors are "small" while the associated σ and RMS values are large). This further highlights the trade-off problem facing standard MMAEs as previously addressed in Chapters 1-3 and test case 1, concerning the difficulty involved in trying to provide accurate state and parameter estimates simultaneously.

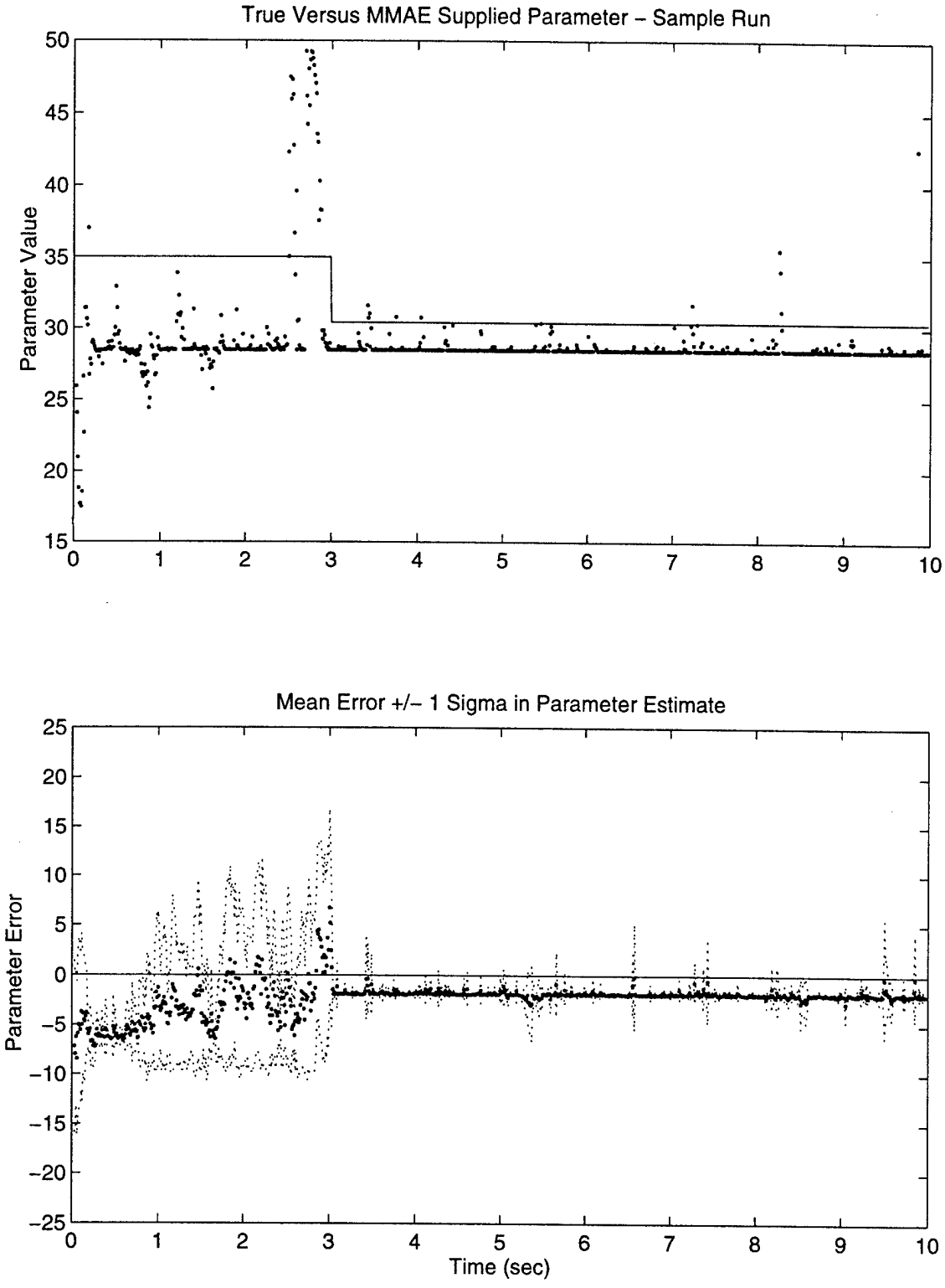


Figure 45. *MMAE/SDPEP* Without IRDF Parameter Estimation Performance: Test Case 8

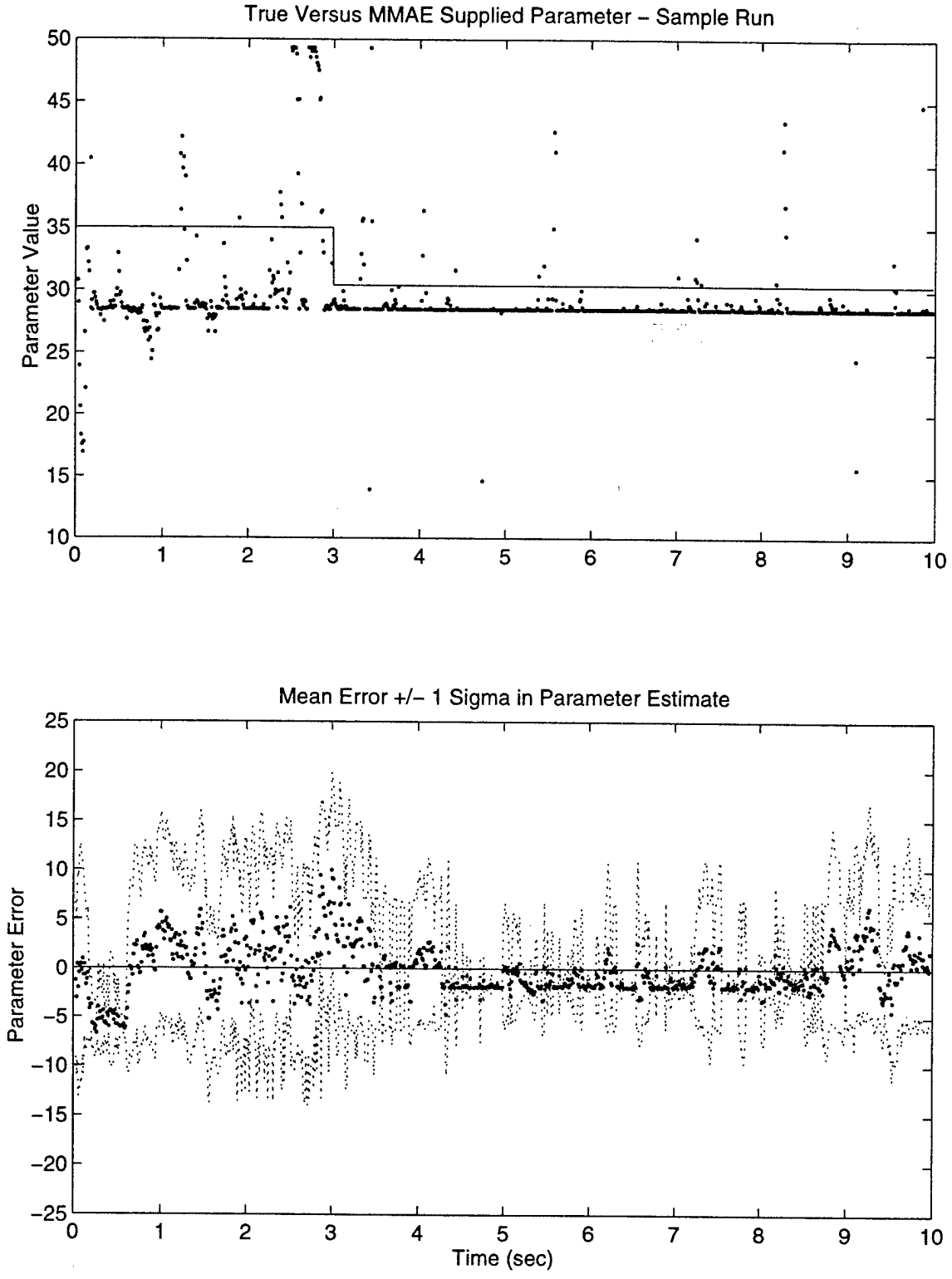


Figure 46. *MMAE/SDPEP* With IRDF Parameter Estimation Performance: Test Case 8

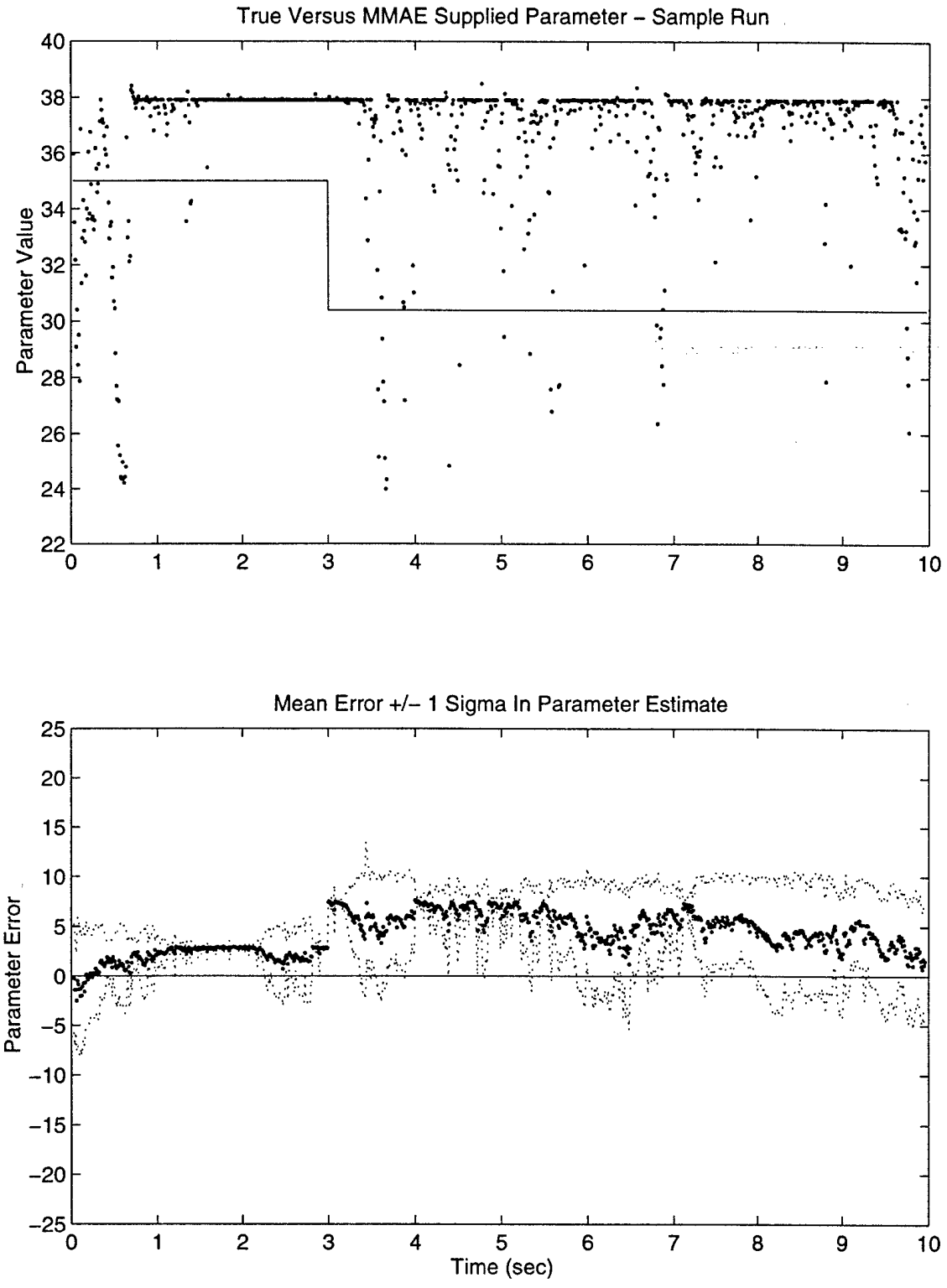


Figure 47. *MMAE/SDSEP* Parameter Estimation Performance: Test Case 8

Summary. Both M^3AEs outperform the conventional *MMAE/SDSEP* in state and parameter estimation performance for this test case. Additionally, note that the approximate M^3AE's covariance analysis shown in Figure 25 on page 120 compares favorably (including predicting the step change) to the actual performance shown in Figure 43, especially at the measurement update times, (t_i^+) (as opposed to the propagate times (t_i^-); however, the overall approximate covariance analysis provides a good upper bound indication of actual performance). Furthermore, this test case shows the potential benefit of the M^3AE architecture in estimating both parameters and states even after undergoing a step change in the true parameter value. Finally, this test case particularly draws attention to the need for removing or reducing the biases in estimating $\hat{\mathbf{a}}$, which will in turn improve the state estimation performance of the M^3AE. Moving-bank M^3AE versus fixed-bank M^3AE will address this issue directly.

4.2 13-State Integrated GPS/INS with Uncertainties in R

The 13-state integrated GPS/INS problem presented below is the same example White researched in his thesis [78]. The integrated GPS/INS system is flown through a simulated tanker flight profile with a mission duration of 3900 seconds. Only 100 seconds of the 3900-second mission is investigated in this example, since it is not the intention of this research to provide an in-depth sensitivity analysis of the M^3AE architecture to all potential real-world flight scenarios from takeoff to landing. The 100 seconds used in the research occur during a straight and level leg of the flight profile from 700 to 800 seconds. During this period, the M^3AE's ability to handle the effects of interference on the GPS signals is investigated. Thus, there is only one uncertain parameter in this problem and it is the variance of the measurement noise affecting the four GPS measurements. Additionally, the majority of the test cases conducted during this example vary a_T stepwise throughout the simulation, representing various levels of interference. Furthermore, IRDF is ineffective in this example and not implemented in two of the three test cases presented. However, a brief discussion addressing its performance is presented in test case 2 to follow. Finally, given the foundation provided in the first example, the presentation format in this section (versus Section 4.1) is as follows:

1. System description – presents models for the INS, radar altimeter, and GPS. Additionally, a description of the expected events (i.e., the GPS signal interference) is presented.

2. Enhanced Sheldon optimization – presents results for both state and parameter discretizations.

3. Simulations and performance analysis – presents analysis of the M^3AE versus the *MMAE/SDSEP* and *MMAE/SDPEP* for each test case. As mentioned above, the impact of IRDF is addressed in the second test case only. Finally, a comparison of the approximate M^3AE covariance analysis tool versus actual performance is presented.

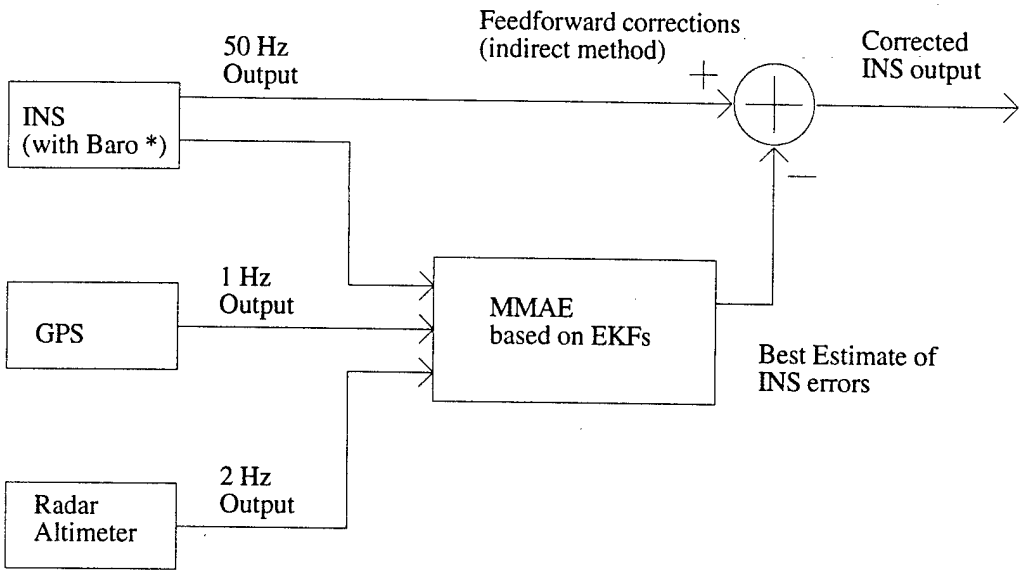
4. Summary – summarizes the major themes discovered during the analysis of the test case.

4.2.1 System Description

The following system description comes directly from [78] with three modifications. In a joint effort with [77], these modifications include reduction of the truth model from 62 to 13 states, deletion of differential GPS to yield conventional GPS, and deletion of the pseudo-lites. The main element of the GPS/INS being monitored for parameter variations in its model is the GPS, specifically with respect to the amount of interference/jamming corruption (measurement noise) that enters the GPS receiver. The INS, barometric altimeter and radar altimeter also provide measurements to the Kalman filter. The following measurements are available: four satellite vehicle (SV) pseudoranges, altitude from the barometric altimeter and height above ground level from the radar altimeter.

A block diagram representing the integrated GPS/INS configuration is shown in Figure 48. The *true* aircraft position is generated by the trajectory profile generator PROFGEN [58] and is provided to the performance evaluation tool during an MMSOFE simulation run. The GPS satellite vehicle (SV) positions are given by actual satellite data recorded on 4 May 1991 and are combined with the true aircraft position to obtain true ranges, which are modified with appropriately modeled noise to provide *pseudoranges* measurements for use by the GPS.

Each navigation system generates measurements that are representable as perturbations from the true range, and the final *difference* measurements are then formed by subtracting the GPS measured ranges from their corresponding INS-calculated ranges. The extended Kalman filter (EKF) equations propagate estimates of the GPS/INS error states and use the measurements to update those state estimates. Finally, these state estimates are used to correct the INS-indicated position at each sample time. The truth *and* filter models consist of 13 states (11 INS states and 2 GPS states) with details to follow.



* Barometric Altimeter incorporated into system to compensate for the fact that a bare INS has an unstable vertical channel

Figure 48. Integrated GPS/INS Block Diagram

4.2.1.1 *INS Models*

The INS Truth and Filter Models. This section presents the truth and filter models used for the INS. The INS is a strapped-down wander azimuth system based on the Litton LN-93. The manufacturer, Litton, developed a 93-state error model [25] describing the error characteristics of the LN-93. The error states $\delta\mathbf{x}$ used in the full model may be separated into 6 categories:

$$\delta\mathbf{x} = \left[\delta\mathbf{x}_1^T \delta\mathbf{x}_2^T \delta\mathbf{x}_3^T \delta\mathbf{x}_4^T \delta\mathbf{x}_5^T \delta\mathbf{x}_6^T\right]^T \tag{177}$$

where $\delta\mathbf{x}$ is a 93-dimensional column vector and:

- $\delta\mathbf{x}_1$ represents the "general" error vector containing 13 position, velocity, attitude, and vertical channel errors; the first nine states are those of the standard Pinson model [65] of INS error characteristics.
- $\delta\mathbf{x}_2$ consists of 16 gyro, accelerometer, and baro-altimeter exponentially time-correlated errors, and "trend" states. These states are modeled as first order Gauss-Markov processes.
- $\delta\mathbf{x}_3$ represents gyro bias errors. These 18 states are modeled as random constants.
- $\delta\mathbf{x}_4$ is composed of accelerometer bias error states. These 22 states are modeled in the same manner as the gyro bias states.
- $\delta\mathbf{x}_5$ depicts accelerometer and gyro initial thermal transients. The 6 thermal transient states are first-order Gauss-Markov processes with $\mathbf{w}_5 = \mathbf{0}$ so as to account for just the initial transient effects.
- $\delta\mathbf{x}_6$ models gyro compliance errors. These 18 error states are modeled as biases.

The original truth model state space differential equation is given by

$$\begin{Bmatrix} \delta\dot{\mathbf{x}}_1 \\ \delta\dot{\mathbf{x}}_2 \\ \delta\dot{\mathbf{x}}_3 \\ \delta\dot{\mathbf{x}}_4 \\ \delta\dot{\mathbf{x}}_5 \\ \delta\dot{\mathbf{x}}_6 \end{Bmatrix} = \begin{bmatrix} \mathbf{F}_{11} & \mathbf{F}_{12} & \mathbf{F}_{13} & \mathbf{F}_{14} & \mathbf{F}_{15} & \mathbf{F}_{16} \\ \mathbf{0} & \mathbf{F}_{22} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{F}_{55} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \delta\mathbf{x}_1 \\ \delta\mathbf{x}_2 \\ \delta\mathbf{x}_3 \\ \delta\mathbf{x}_4 \\ \delta\mathbf{x}_5 \\ \delta\mathbf{x}_6 \end{Bmatrix} + \begin{Bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{Bmatrix} \tag{178}$$

This 93-state error model is a highly accurate LN-93 representation, but the high dimensionality of the state equation makes the model prohibitively CPU-intensive (computationally, and in terms of storage) for projects examining a large number of problem variations. The work of Negast [61] addressed the reduction of the INS error-state model while preserving enough fidelity to be considered a viable truth model.

A reduced-order model is used for both the truth and filter model in this research and is defined in Equation (179):

$$\begin{Bmatrix} \delta\dot{\mathbf{x}}_1 \\ \delta\dot{\mathbf{x}}_2 \end{Bmatrix} = \begin{bmatrix} \mathbf{F}_{(red)11} & \mathbf{F}_{(red)12} \\ \mathbf{0} & \mathbf{F}_{(red)22} \end{bmatrix} \begin{Bmatrix} \delta\mathbf{x}_1 \\ \delta\mathbf{x}_2 \end{Bmatrix} + \begin{Bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{Bmatrix} \tag{179}$$

Note that the submatrix indices used in representing the 13-state model are not identical to those used in outlining the 93-state INS error model. This difference is indicated by the notation $\mathbf{F}_{(red)}$ for reduced order. The relationship between the two models is shown in Appendix A. Therefore, the INS filter model is comprised of 11 states (the first nine being the standard Pinson error model states): 3 platform misalignment errors, 3 velocity errors, 3 position errors, and 2 states for barometric altimeter stabilization.

The INS Measurement Model. The only measurement model associated directly with the INS is that for barometric altimeter aiding. The altimeter aiding is used to compensate for the instability inherent in the vertical channel of the INS. The altimeter output Alt_{Baro} is modeled as the sum of the true altitude h_t, the error in the barometric altimeter δh_B, and a random measurement noise v of variance $R_{Baro} = 3500\ ft^2$. Similarly, the INS-calculated altitude Alt_{INS} is the sum of the true altitude and the INS error in vehicle altitude above the reference ellipsoid, δh. A *difference* measurement is used to eliminate the unknown true altitude, h_t, resulting in Equation (180):

$$\begin{aligned} \delta z &= Alt_{INS} - Alt_{Baro} \\ &= [h_t + \delta h] - [h_t + \delta h_B - v] \\ &= \delta h - \delta h_B + v \end{aligned} \tag{180}$$

INS error in vehicle altitude above the reference ellipsoid, δh, and total barometric altimeter correlated error, δh_B, are states 10 and 11 in the 11-state INS model. See Appendix A for the models and numerical values of the model parameters.

4.2.1.2 The Radar Altimeter Model

A radar altimeter is incorporated into this application because of the intent of generating a precision landing system. A GPS-aided baro-inertial system does not have sufficient accuracy in the vertical direction for this purpose, so the additional accurate input from a radar altimeter is used. The measurement equation of the radar altimeter is based on the difference between the INS-predicted altitude Alt_{INS} and the radar altimeter measurement Alt_{Ralt}:

$$\begin{aligned} \delta z &= Alt_{INS} - Alt_{Ralt} \\ &= [h_t + \delta h] - [h_t - v] \\ &= \delta h + v \end{aligned} \tag{181}$$

The errors in the radar altimeter are modeled as white noise with no time-correlated component. This may be a rather crude model, but should be sufficient to demonstrate performance trends. Note that no additional states are required with the addition of this radar altimeter model.

The radar altimeter measurement noise variance R_{Ralt} is a function of aircraft altitude above ground level (AGL) and will be the same in the truth and filter models. The radar altimeter noise's altitude-dependent variance [24] is given by

$$R_{Ralt} = \left\{[0.01]^2 * [\mathrm{AGL}_{true}]^2\right\} + 0.25\ ft^2 \tag{182}$$

4.2.1.3 GPS Models

The GPS generates user position based on "known" ranges to satellites at "known" positions. The satellites themselves transmit their position in space (in the form of ephemeris data) as accurately as it is known and the exact time (also a best estimate) at which the transmission is sent. The

actual range information is calculated based on knowledge of the satellite position and the finite propagation speed of the electromagnetic radiation emitted from the satellite.

The 30-State GPS System Model. The GPS model used in this work was developed by past researchers at AFIT [16, 61, 76]. The dynamics and measurement equations for the full 30-state system model are presented in this section. Five types of error sources are modeled in the GPS state equations. The first error type, user clock error, is common to all SV's. The remaining four error types are unique to each SV. The first two states represent user clock errors and are modeled as:

$$\begin{Bmatrix} \dot{x}_{Uclk_b} \\ \dot{x}_{Uclk_{dr}} \end{Bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{Bmatrix} x_{Uclk_b} \\ x_{Uclk_{dr}} \end{Bmatrix} \tag{183}$$

where

x_{Uclk_b} = range equivalent of user clock bias
$x_{Uclk_{dr}}$ = velocity equivalent of user clock drift

The initial state estimates and covariances for these states were chosen to be consistent with previous AFIT research [7, 16, 61, 76] and are:

$$\begin{Bmatrix} \widehat{x}_{Uclk_b}(t_0) \\ \widehat{x}_{Uclk_{dr}}(t_0) \end{Bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{184}$$

and

$$\mathbf{P}_{Uclk_b,Uclk_{dr}}(t_0) = \begin{bmatrix} 9.0 \times 10^{14}\ ft^2 & 0 \\ 0 & 9.0 \times 10^{10}\ ft^2/sec^2 \end{bmatrix} \tag{185}$$

Because these error sources are a function of the user equipment, they are common to all the SV's. Recall that each of the remaining error types is specific to each SV, denoted by a subscript j.

The second error type is the code loop error δPR_{cloop_j}. The code loop is part of the user equipment shared by all the SV's, but its error magnitude is relative to each SV. The third GPS

error type is the result of atmospheric interference with the electromagnetic (EM) signals broadcast by each SV, specifically, ionospheric and tropospheric delay, δPR_{ion_j}, and δPR_{trop_j}. The code loop error, tropospheric delay, and ionospheric delay are all modeled as first-order Gauss-Markov processes with time constants shown in Equation (186). All three are driven by zero-mean white Gaussian noise with strength shown in Equation (189). The fourth error source is due to inaccuracies of the clocks on board the individual SV's, δPR_{Sclk_j}. The final GPS error source is based on line-of-sight errors between the SV's and the receiver, δx_{s_j}, δy_{s_j}, and δz_{s_j}.

$$\begin{Bmatrix} \delta\dot{P}R_{cl_j} \\ \delta\dot{P}R_{trop_j} \\ \delta\dot{P}R_{ion_j} \\ \delta\dot{P}R_{Sclk_j} \\ \delta\dot{x}_{s_j} \\ \delta\dot{y}_{s_j} \\ \delta\dot{z}_{s_j} \end{Bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{500} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{1500} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} \delta PR_{cl_j} \\ \delta PR_{trop_j} \\ \delta PR_{ion_j} \\ \delta PR_{Sclk_j} \\ \delta x_{s_j} \\ \delta y_{s_j} \\ \delta z_{s_j} \end{Bmatrix} + \begin{Bmatrix} w_{cl_j} \\ w_{trop_j} \\ w_{ion_j} \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix} \tag{186}$$

with initial covariance values given by

$$\mathbf{P}_{GPS} = \begin{bmatrix} 0.25\ ft^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.0\ ft^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.0\ ft^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25\ ft^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25\ ft^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 25\ ft^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 25\ ft^2 \end{bmatrix} \tag{187}$$

(where all but the $\mathbf{P}_{GPS}(3,3)$ term have stationary characteristics. Though this value may seem strange, it was taken directly from [61]) and noise means and strengths given by

$$E[\mathbf{w}_{GPS}(t)] = \mathbf{0} \tag{188}$$

$$E[\mathbf{w}_{GPS}(t)\mathbf{w}_{GPS}^T(t+\tau)] = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.004 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.004 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} ft^2/sec \cdot \delta(\tau) \tag{189}$$

The full 30-state GPS dynamics matrix is not shown explicitly but may be easily constructed by augmenting Equation (183) and four copies (one for each SV) of Equation (186).

The GPS Truth Model and Filter Design Models. Research has shown [56,61] that the two user clock error states provide a sufficient filter model for GPS. The primary argument is that the errors modeled for the 28 other GPS states (assuming four SV's) are small when compared to the user clock errors which are common to all SV's. By increasing the dynamics driving noise and re-tuning the filter, the overall performance of the integrated navigation system can be maintained. The GPS truth and filter models used in this research are given by Equation (183) plus noise:

$$\begin{Bmatrix} \dot{x}_{Uclk_b} \\ \dot{x}_{Uclk_{dr}} \end{Bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{Bmatrix} x_{Uclk_b} \\ x_{Uclk_{dr}} \end{Bmatrix} + \begin{Bmatrix} w_{clk_b} \\ w_{clk_{dr}} \end{Bmatrix} \tag{190}$$

The GPS Measurement Model. The pseudorange measurements available to the GPS receiver are the sum of the true range, several error sources, and a random noise:

$$PR_{GPS_j} = PR_{t_j} + \delta PR_{cloop_j} + \delta PR_{trop_j} + \delta PR_{ion_j} + \delta PR_{Sclk_j} + \delta PR_{Uclk} - v_j \tag{191}$$

where

PR_{GPS_j}	=	GPS pseudorange measurement, from SV$_j$ to user
PR_{t_j}	=	true range, from SV$_j$ to user
δPR_{cloop_j}	=	range error due to code loop error
δPR_{trop_j}	=	range error due to tropospheric delay
δPR_{ion_j}	=	range error due to ionospheric delay
δPR_{Sclk_j}	=	range error due to SV$_j$ clock error
δPR_{Uclk}	=	range error due to user clock error
v_j	=	zero-mean white Gaussian measurement noise, $R_j = 9\ ft^2$

Because PR_t is not available to the filter, the difference between GPS pseudorange and INS-indicated pseudorange will be taken eventually to eliminate this term. First, the satellite position vector $\mathbf{X}_S$ and the user position vector $\mathbf{X}_U$ are defined as:

$$\mathbf{X}_U = \left\{ \begin{array}{c} x_u \\ y_u \\ z_u \end{array} \right\}^e \quad , \quad \mathbf{X}_S = \left\{ \begin{array}{c} x_s \\ y_s \\ z_s \end{array} \right\}^e \tag{192}$$

where the superscript e denotes coordinates in the earth-centered earth-fixed (ECEF) frame. The pseudorange from the user to the satellites calculated by the INS, PR_{INS}, is the difference between the GPS/INS-calculated user position, $\mathbf{X}_U$, and the satellite position given by the ephemeris data, $\mathbf{X}_S$:

$$PR_{INS} \;=\; |\mathbf{X}_U - \mathbf{X}_S| = \left| \left\{ \begin{array}{c} x_U \\ y_U \\ z_U \end{array} \right\}^e - \left\{ \begin{array}{c} x_S \\ y_S \\ z_S \end{array} \right\}^e \right| \tag{193}$$

An equivalent form of Equation (193) is:

$$PR_{INS} = \sqrt{(x_U - x_S)^2 + (y_U - y_S)^2 + (z_U - z_S)^2} \tag{194}$$

With perturbations representing errors in $\mathbf{X}_U$ and $\mathbf{X}_S$, Equation (194) can be written in terms of the true range via a truncated first-order Taylor series:

$$\begin{aligned} PR_{INS} = \;& PR_t + \left. \frac{\partial PR_{INS}(\mathbf{X}_S, \mathbf{X}_U)}{\partial \mathbf{X}_S} \right|_{(\mathbf{X}_S, \mathbf{X}_U)_{nom}} \cdot \delta \mathbf{X}_S \\ & + \left. \frac{\partial PR_{INS}(\mathbf{X}_S, \mathbf{X}_U)}{\partial \mathbf{X}_U} \right|_{(\mathbf{X}_S, \mathbf{X}_U)_{nom}} \cdot \delta \mathbf{X}_U \end{aligned} \tag{195}$$

The solution for PR_{INS} is found by evaluating the partial derivatives of Equation (194) to get:

$$
\begin{aligned}
PR_{INS} = \; & PR_t - \left[\tfrac{x_S - x_U}{|PR_{INS}|}\right] \cdot \delta x_U - \left[\tfrac{y_S - y_U}{|PR_{INS}|}\right] \cdot \delta y_U - \left[\tfrac{z_S - z_U}{|PR_{INS}|}\right] \cdot \delta z_U \\
& + \left[\tfrac{x_S - x_U}{|PR_{INS}|}\right] \cdot \delta x_S + \left[\tfrac{y_S - y_U}{|PR_{INS}|}\right] \cdot \delta y_S + \left[\tfrac{z_S - z_U}{|PR_{INS}|}\right] \cdot \delta z_S
\end{aligned}
\tag{196}
$$

Finally, the truth model GPS pseudorange *difference* measurement is given as:

$$
\begin{aligned}
\delta z = \; & PR_{INS} - PR_{GPS} \\
= \; & - \left[\tfrac{x_S - x_U}{|PR_{INS}|}\right] \cdot \delta x_U - \left[\tfrac{y_S - y_U}{|PR_{INS}|}\right] \cdot \delta y_U - \left[\tfrac{z_S - z_U}{|PR_{INS}|}\right] \cdot \delta z_U \\
& + \left[\tfrac{x_S - x_U}{|PR_{INS}|}\right] \cdot \delta x_S + \left[\tfrac{y_S - y_U}{|PR_{INS}|}\right] \cdot \delta y_S + \left[\tfrac{z_S - z_U}{|PR_{INS}|}\right] \cdot \delta z_S \\
& - \delta PR_{cloop} - \delta PR_{trop} - \delta PR_{ion} - \delta PR_{Sclk} - \delta PR_{Uclk} + v
\end{aligned}
\tag{197}
$$

The user position errors in Equation (197) can be derived from the first three (position error) states of the filter or truth model using an orthogonal transformation [6].

The reduced order truth and filter design measurement models for the GPS measurement do not contain terms for the errors due to code loop variations, atmospheric delays, satellite clock deviations, or errors in ephemeris-given satellite position. The filter GPS measurement model can be written as:

$$
\delta z = - \left[\frac{x_S - x_U}{|PR_{INS}|}\right] \cdot \delta x_U - \left[\frac{y_S - y_U}{|PR_{INS}|}\right] \cdot \delta y_U - \left[\frac{z_S - z_U}{|PR_{INS}|}\right] \cdot \delta z_U - \delta PR_{Uclk} + v
\tag{198}
$$

4.2.1.4 Event Models

This section discusses the methods used to model the event changes in the MMSOFE simulations. Interference/jamming is modeled as a sudden increase in the measurement noise associated with all four SV's, resulting in lower carrier-to-noise ratios, C/N_0, in the GPS receiver. This event is induced in all SV measurements because interference/jamming is assumed to occur at the receiver,

which will affect all four channels simultaneously. The interference noise variance, R_{int}, is added to the truth model measurements' R_j values (see discussion of Equation (191)) to simulate real-world interference and will be allowed to take on selected values within the interference parameter space spanned by the MMAE filter bank. Emphasis will be placed on demonstrating the capability of MMAE to detect and identify interference events of unspecified magnitude quickly. GPS *jamming* is used to refer to the total loss of useful GPS transmissions due to very large signal interference. A GPS jamming event is well-modelled (and much more easily modelled) via *very large* measurement noise. When the MMAE algorithm detects very large diagonal elements in real-world measurement noise covariance matrix $\mathbf{R}$, then the corresponding measurements will be *very* lightly weighted by the elemental Kalman filters; the effect is essentially the same as if those measurements were never received, hence the use of the term "interference/jamming." The system measurement noise covariance matrix is given by:

$$\mathbf{R} = \begin{bmatrix} R_{Ralt} & & & & & \\ & R_{GPS} & & & \mathbf{0} & \\ & & R_{GPS} & & & \\ & & & R_{GPS} & & \\ & \mathbf{0} & & & R_{GPS} & \\ & & & & & R_{Baro} \end{bmatrix} \tag{199}$$

where R_{GPS} is the product of the nominal GPS measurement noise variance, R_0, and either a true multiplier, a_t, or a filter-assumed multiplier, a_j.

$$R_{GPS} = \begin{cases} a_t R_0 & , \text{ for truth model} \\ a_j R_0 & , \text{ for filter models} \end{cases} \tag{200}$$

Previous research [61, 76, 78] determined reasonable choices for the nominal GPS measurement noise and the range for the noise multiplier as:

$$\begin{aligned} R_0 &= 9\ ft^2 \\ 1 &\leq a_{t,j} \leq 2000 \end{aligned} \tag{201}$$

Since the GPS measurement noise will be chosen as the scalar parameter for estimation, future notation will indicate values for the minimum and maximum allowable parameter values as:

$$a_{min} = 1$$
$$a_{max} = 2000$$

Table 14 show the three test cases used in the performance analysis. The entries in the table represent the interference levels (multiplier values on the four diagonal terms of $\mathbf{R}_T$ corresponding to the GPS measurements) for each case and the time in seconds when a parameter change occurs (this will be discussed further in Section 4.2.3)

Table 14. Simulated Test Cases (Interference Noise Variance Multiplier Levels)

Time	Case 1 a_t value:	Case 2 a_t value:	Case 3 a_t value:
700	1	1	1000
718	1	750	500
760	1	1	1

4.2.2 Sheldon Parameter Optimization Algorithm

Recall that Sheldon's parameter optimization algorithm (see Sections 2.2 and 3.5 for algorithm details) is implemented in MATLAB [40]. For this 13-state nonlinear, unstable system example, only the constrained-range optimization (using "*constr*" from MATLAB's Optimization Toolbox [39]) using a 20-sample finite horizon (20-samples represents 20 seconds for this problem, which provides a reasonable projection into the future for an aircraft navigation system flying straight-and-level) is implemented to determine the appropriate elemental filter parameter values. Additionally, for the test cases examined in this example, a lower bound ($p_{min} = 0.001$ chosen quite small, due to the large range of the parameter space and the very coarse discretization of the parameter space so that large probabilities are not artificially assigned to very poor parameter/state estimates by such

lower bounding) is placed on the elemental filter probabilities. Therefore, Equation (59) is used to solve for the $\hat{a}(t_i)$ used in the cost function in Equation (46).

The results of conducting the enhanced Sheldon's constrained-range optimization, with the 20-sample finite horizon, is shown in Table 15, where the unknown parameter is the variance of the measurement noise, R_{GPS} in Equations (199) and (200). Additionally, Figure 49 displays the results of the constrained-range optimization used in the parameter estimation case. The range of the optimization was from R_{GPS}= 9 ft^2 to 18000 ft^2. The minimum value, $R_0 = 9\ ft^2$, represents "normal" operation when no interference to the GPS signal is occurring. The maximum value, $R_{\max} = 18000\ ft^2$, represents an increase in the measurement noise by a factor of 2000 (i.e., measurement noise $R_{\max} = R_0 * 2000$), and is assumed to be the highest level of real-world interference that will be encountered. The valleys in the curve provide the optimal locations for placing the elemental filter $\mathbf{a}_j$'s. Although difficult to see on Figure 49, the first minimum occurs at $R = 9$ as in Table 15, and the next two minima are also given in Table 15 for the case of optimizing for parameter estimation performance. As discussed in the 2-state example, if $\mathbf{a}_T$ is *near* any of the peaks in the curve, it is anticipated that parameter and state estimation performance will be poor [69]. Also recall from Section 4.1, that the following three abbreviations are used for convenience: *closest* = "closest" in the "Baram distance measure sense"; *MMAE/SDSEP* = MMAE "Sheldon-discretized" for state estimation performance; and *MMAE/SDPEP* = MMAE "Sheldon-discretized" for parameter estimation performance.

Table 15. Parameter Choices for State and Parameter Estimation

Filter	State Estimation Case	Parameter Estimation Case
1	9	9
2	8176	3998
3	17369	12578

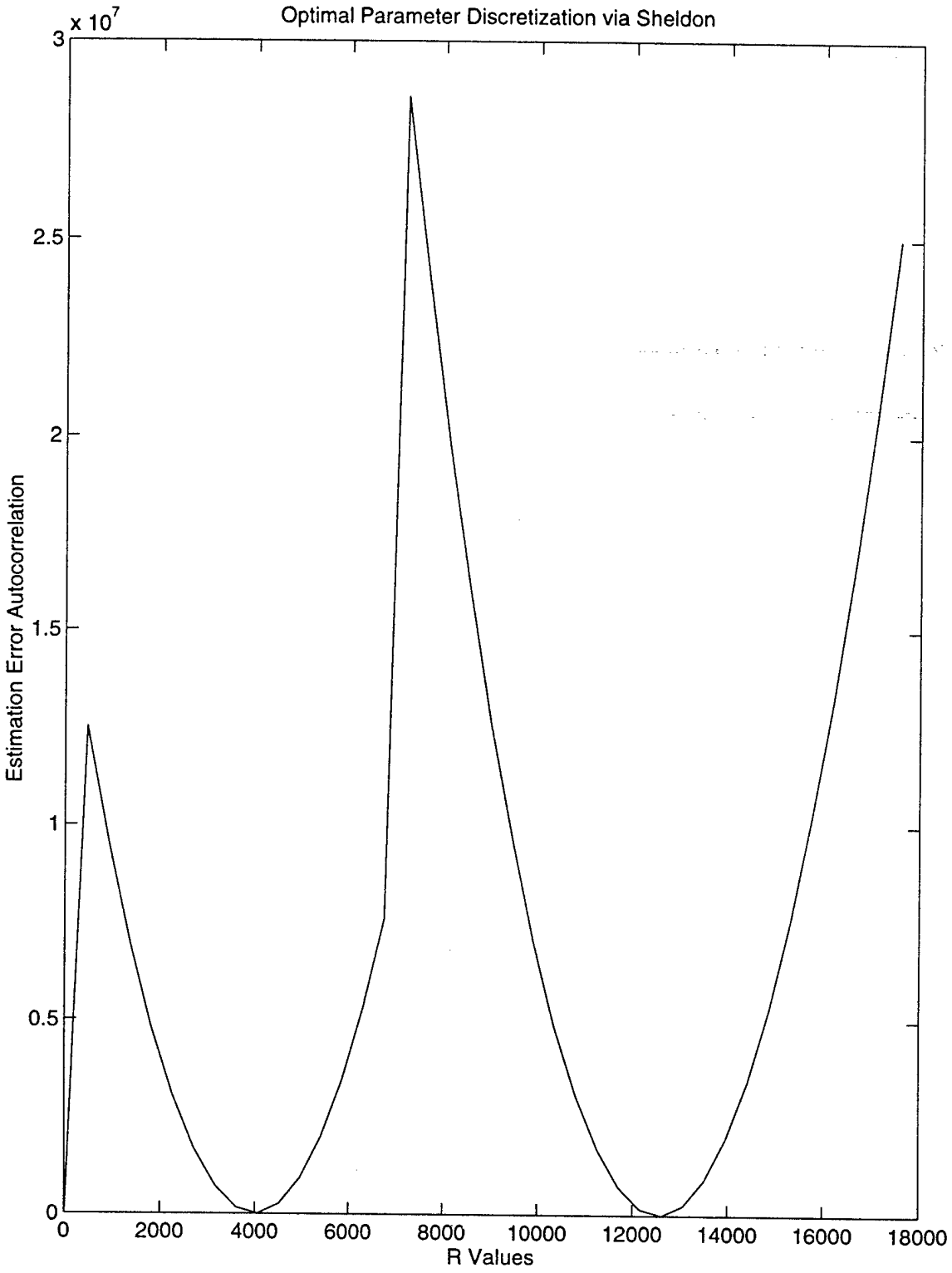


Figure 49. Autocorrelation Curve For Constrained-Range Parameter Discretization - Example 2

Notice that in both the state and parameter estimation cases, the first elemental filters have the same value. These are the filters which are tuned for "normal" operation when no interference is occurring. Since this represents "normal" operation, it was decided to *force* one of the elemental filters in either MMAE to be based on this parameter value. Therefore, after "fixing" the value of the first elemental filter, each constrained-range optimization algorithm (for either state or parameter discretization) determines the remaining two filters' parameter values based upon which of the two optimization algorithms are implemented.

4.2.3 Simulations and Performance Analysis

Three test cases are conducted to compare performance between the M^{3}AE and a conventional fixed-bank *MMAE/SDSEP*. The three test cases are listed in Table 16, which shows the actual magnitudes of the measurement noise seen during the simulation. This is simply a repeat of Table 14, with the entries multiplied by $R_0 = 9\ ft^2$. Additional test cases were conducted, but provided no further insight into the M^{3}AE's performance, and thus are not presented or discussed.

Only the first test case keeps the parameter fixed throughout the simulation (representing *normal* operation), while the remaining two cases experience step changes (sudden increases or decreases in the measurement noise due to interference) in the true parameter value as shown in Table 16.

Table 16. M^{3}AE Test Cases: True R_{GPS} Values (ft^2)

Time	Case 1	Case 2	Case 3
700	9	9	9000
718	9	6750	4500
760	9	9	9

Recall from the first example, that the M^{3}AE provides the greatest parameter and state estimation performance benefit over the *MMAE/SDSEP* when "good" blending of the MMAE elemental filters' parameter estimates is achieved ("good" blending is defined as blending that produces a near-zero-mean-error parameter estimate). Given that this 13-state GPS/INS example uses only three elemental filters and a coarse discretization, any MMAE "blending" of state and parameter estimates is difficult to achieve. Of all the test cases actually investigated, test case 2 best demonstrates the case when "good" blending occurs, and highlights the strength of the M^{3}AE architecture. Finally, test case 3 is intended to demonstrate some aspects of the ability of the M^{3}AE to handle multiple step changes in R_{GPS} and also to emphasize that, without "good" blending, the M^{3}AE doesn't show significant improvement over a stand-alone MMAE.

In each test case, a 10-run Monte Carlo simulation is conducted on the following two architectures:

1. An MMAE "Sheldon-discretized" for optimized state estimation performance
2. An M^{3}AE implemented without IRDF, with its internal MMAE "Sheldon-discretized" for optimized parameter estimation performance.

Additionally, in the second test case only, an additional 10-run Monte Carlo simulation is conducted on an M^{3}AE implemented with discrete-time IRDF. IRDF is not implemented in each test case, since for this example problem, Lund's IRDF technique provided no additional performance benefit, as will be shown in the analysis in test case 2.

A comparative analysis is accomplished between these architectures. The same types of plots used in the first example (see Section 4.1.3) are presented here to aid in the analysis. Only the three position states are plotted (i.e., aircraft latitude, longitude, and altitude). The top plots in the state estimation performance figures indicate the MMAEs' and M^{3}AE's state estimation performance in

latitude, the middle plots indicate the state estimation performance in longitude, and the bottom plots indicate the state estimation performance in the altitude channel. Each plot contains 5 curves representing the following:

- The single solid line (which is generally located between all the other traces) represents the actual 10-run mean value for the error in the estimate of each state variable.
- The two other solid traces represents the zero $\pm$ one filter-predicted sigma bound for the error in each state variable's estimate.
- The dot/dashed line represents the 10-run mean $\pm$ one sigma residual for the error in the estimate of each state variable.

The top plots in parameter estimation figures present the parameter estimate performance (dotted line) versus the true parameter value (the solid line) for a representative sample run from the 10-run Monte Carlo simulation. The bottom plots represent the mean error (solid line) and the mean ± 1 standard deviation (dotted line) values from the 10-run Monte Carlo simulation.

Table 17 summarizes the plots generated for each test case versus the architectures listed above. The figure numbers associated with each case are listed in the columns.

Table 17. Summary of Plots

Architecture	MMAE Blended State Est.	M^3AE State Est.	Parameter Estimation
M^3AE without IRDF	50, 56, 66	52, 58, 68	53, 60, 69
M^3AE with IRDF	62	63	64
MMAE/SDSEP	51, 57, 67		54, 61, 70

As in the first example, a table summarizes the temporally averaged RMS values of the state estimation errors. Recall that the plots should be used as the primary indicator of performance, whereas the temporally averaged RMS values provide only a gross indication of state estimation performance, since any amount of high variance data may adversely skew the results. Finally, the

results of the approximate M^{3}AE covariance analysis is presented after discussion of the M^{3}AE's state and parameter estimation performance, for each of the three test cases.

4.2.3.1 Test Case 1: $a_T = 9.0$

In this test case, the true parameter value, $a_T = 9.0$, is fixed throughout the simulation. This case represents a *normal* operating environment in which no interference is occurring. Also, notice that each MMAE's first elemental filter parameter value, a_1, matches the true parameter value, $a_T = 9.0$. An analysis of the actual state and parameter estimation performance is presented next, followed by a comparison of the performance of the M^{3}AE's approximate covariance analysis versus actual M^{3}AE performance.

State Estimation Performance. Figures 50 and 51 show the plots from each MMAE's blended state estimation performance. Figure 52 shows the M^{3}AE state estimation performance. The temporally averaged RMS state estimation errors from each plot are summarized in Table 18 below.

Table 18. Test Case 1: Temporally Averged RMS State Estimation Errors (ft)

State	*MMAE/SDPEP* without IRDF	Conventional *MMAE/SDSEP*	M^{3}AE without IRDF
Lat	1.224	1.224	0.8897
Long	1.189	1.189	0.8969
Alt	2.839	2.838	2.319

In this test case, the figures and the table indicate that all the MMAEs perform the same during the simulation. This is anticipated since each MMAE has the same parameter value, a_1, in its first elemental filter. Therefore, all but 0.002 of the probability weight is given to the first elemental filter in each MMAE which accounts for the nearly identical state estimation performance. Additionally, notice that the M^{3}AE's state estimation performance is essentially the same as that of the

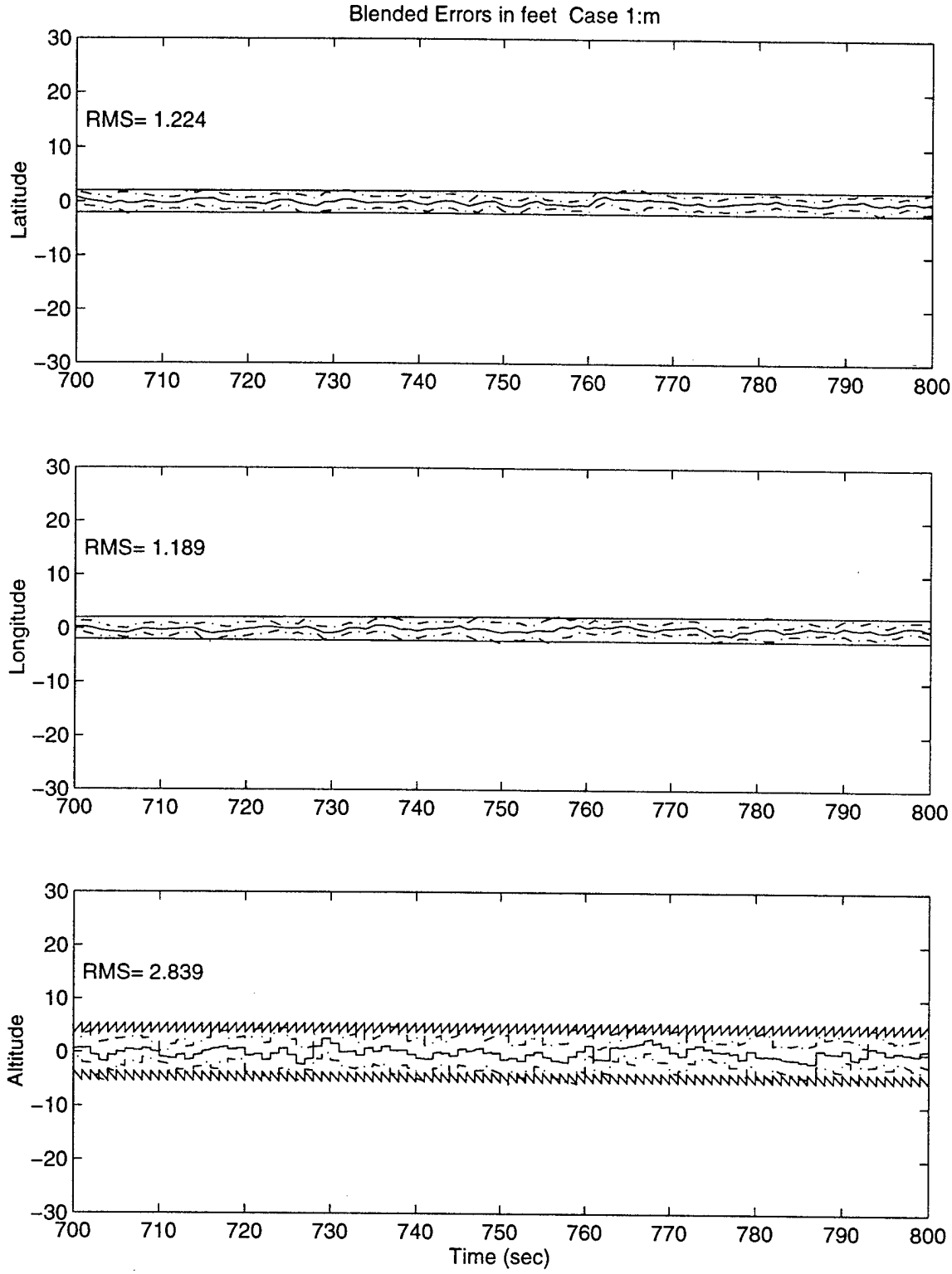


Figure 50. *MMAE/SDPEP* Without IRDF State Estimation Performance: Test Case 1

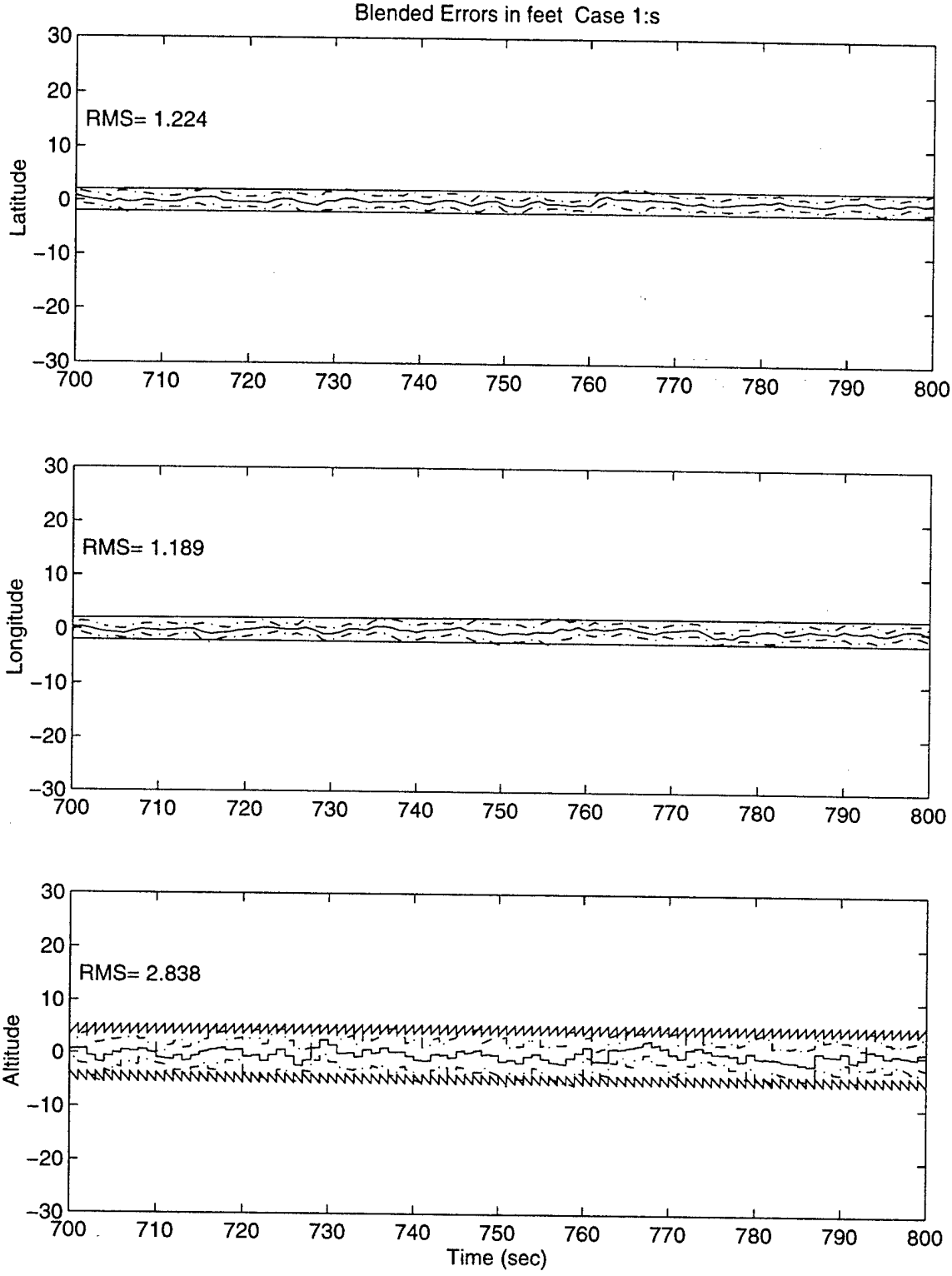


Figure 51. *MMAE/SDSEP* State Estimation Performance: Test Case 1

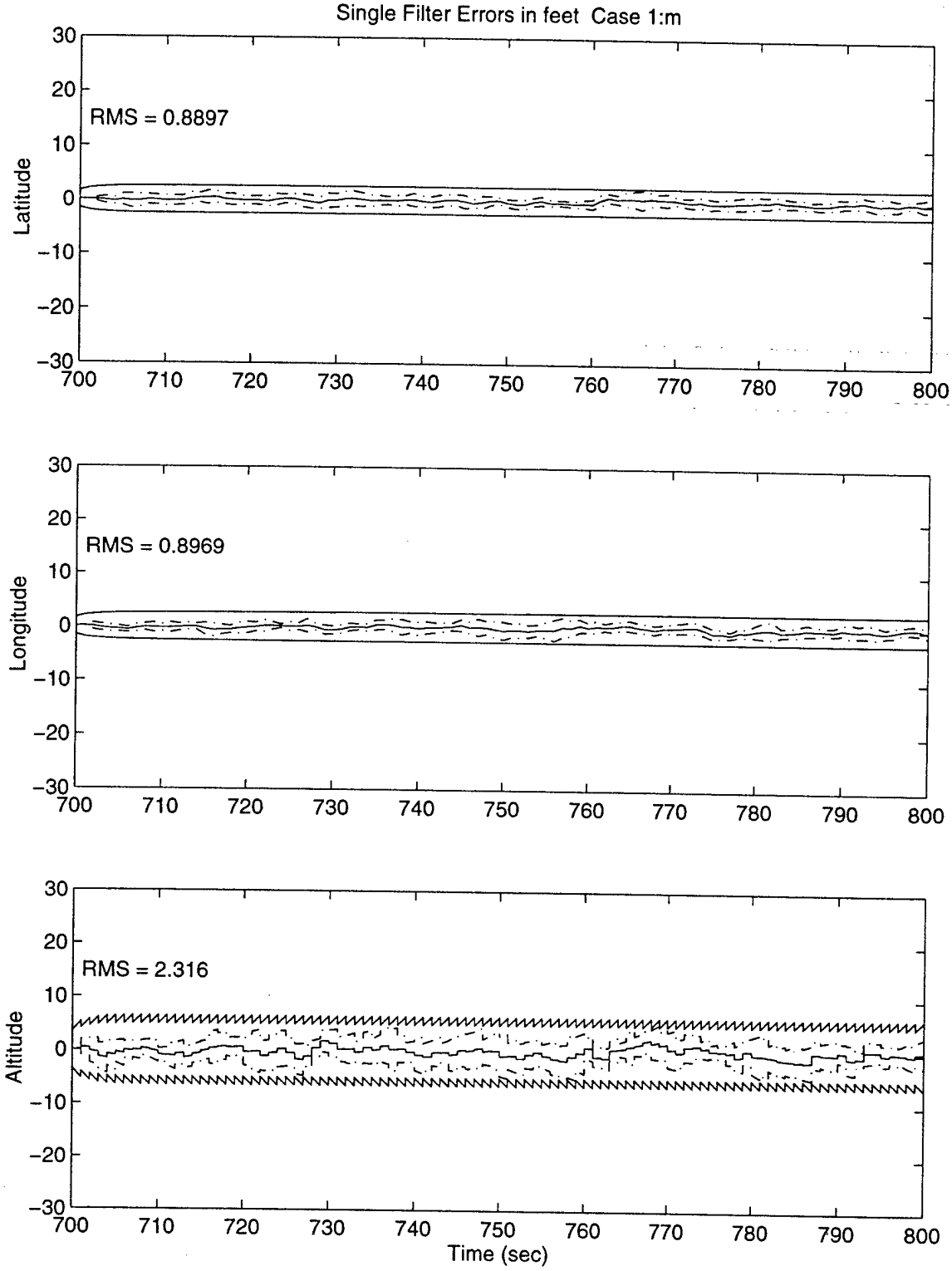


Figure 52. M^{3}AE Without IRDF State Estimation Performance: Test Case 1

MMAEs, which is anticipated since no "good" blending of the estimates is occurring. A comparison of the plots confirm that the state estimation performance is similar between the architectures. Furthermore, notice that the filter-predicted sigma performance in the M^{3}AE is slightly larger than the *MMAE/SDPEP*. This is a direct result of providing the M^{3}AE's single filter a biased-high $\mathbf{R}$ value. Finally, the initial transients shown in the M^{3}AE state estimation plots are due to the fact the actual simulation is started at time equal to 700 seconds, with initial conditions set to zero (this could be corrected by giving proper initial conditions or starting the simulation earlier) and should be ignored in performance comparisons.

Parameter Estimation Performance. Figures 53 and 54 show the plots of each MMAE's blended parameter estimation performance. Notice in both configurations that the parameter estimate is biased. This is a direct result of applying the minimum probability technique to the MMAEs. The resulting bias is exactly equal to:

$$bias = (0.998a_1 + 0.001a_2 + .001a_3) - a_{\mathrm{T}} \tag{202}$$

The bias is larger in the *MMAE/SDSEP* versus the *MMAE/SDPEP* because the discrete parameter values in the *MMAE/SDSEP* are larger than the corresponding discrete parameter values in the *MMAE/SDPEP*. This attribute has caused some researchers [44, 46, 50, 73]to impose the lower bounds on $p_j(t_i)$ computations within an MMAE (to preclude "lockout" as discussed in Section 2.1), but then to set those $p_j(t_i)$ values to zero in Equation (40) or (41) (and then rescaling the remaining $p_j(t_i)$'s to add to one) for enhanced performance.

M^{3}AE Approximate Covariance Analysis. Figure 55 shows the results of the approximate covariance analysis (the three seconds of missing data at the end of this plot is a function of the software code and occurs in all the covariance analysis plots shown in this example). In this test case the M^{3}AE approximate covariance analysis compares closely with the M^{3}AE's single-filter-predicted σ values (versus the true σ values) of Figure 52, but only indicates a performance upper

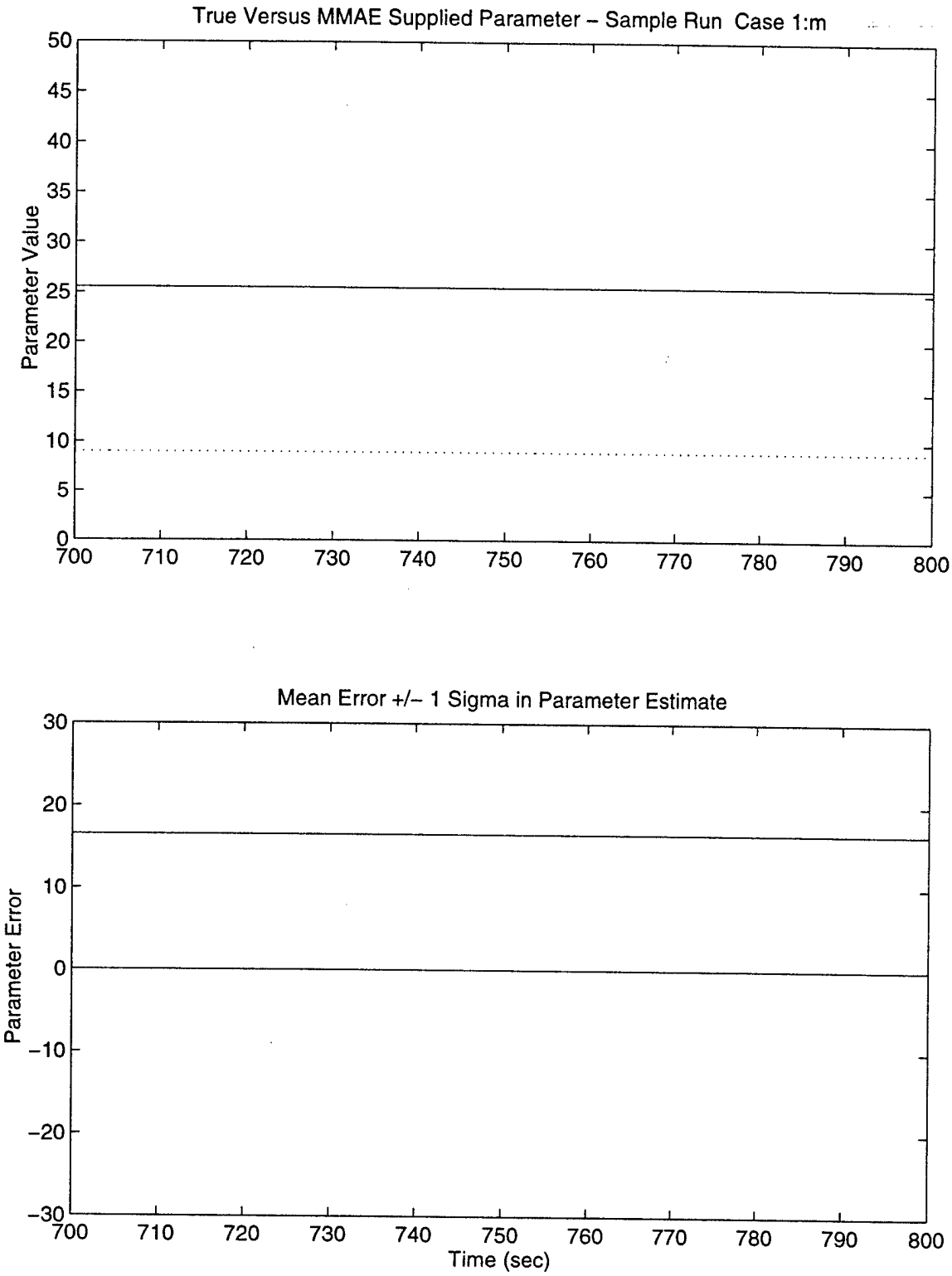


Figure 53. *MMAE/SDPEP* Without IRDF Parameter Estimation Performance: Test Case 1

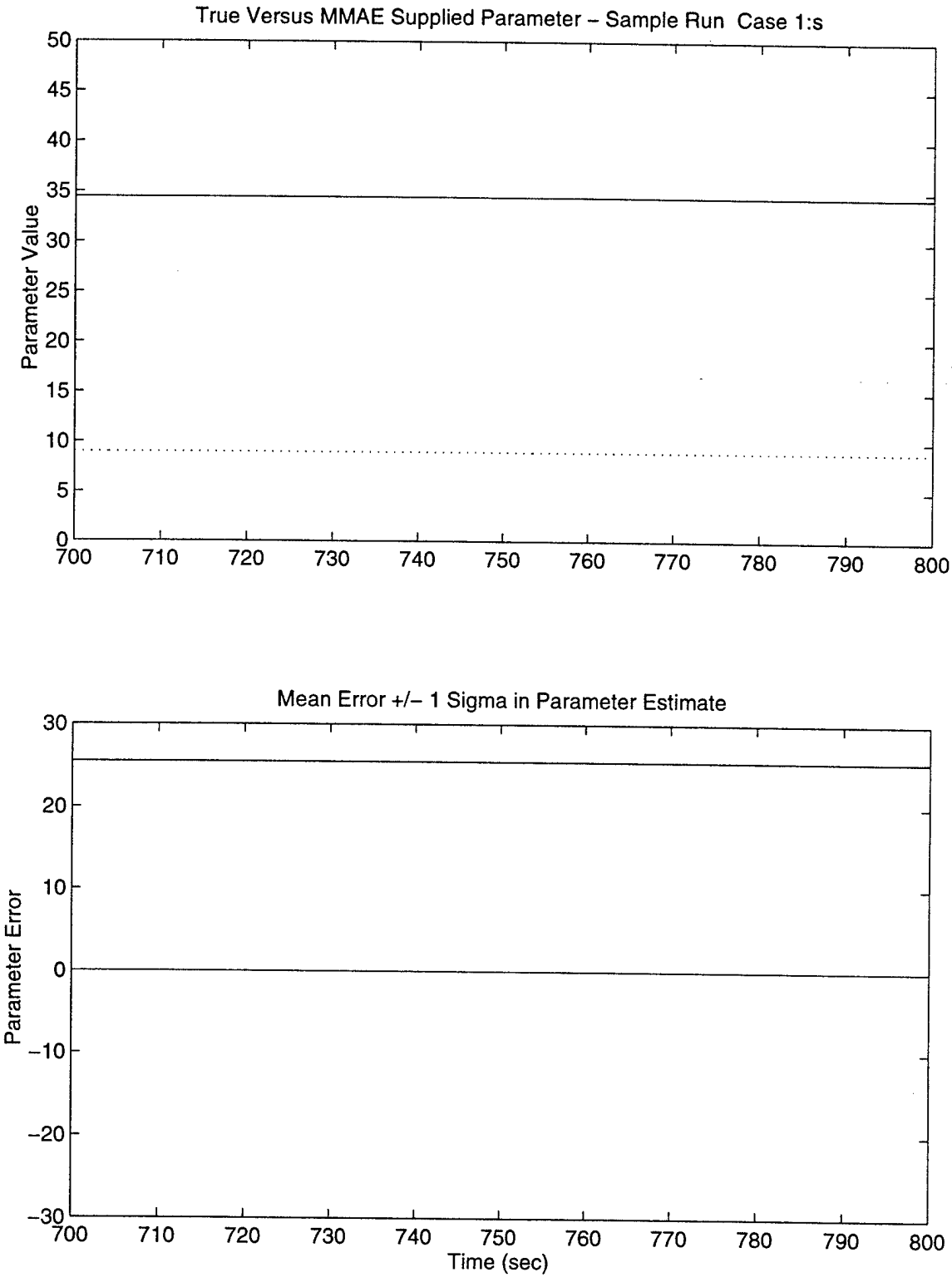


Figure 54. *MMAE/SDSEP* Parameter Estimation Performance: Test Case 1

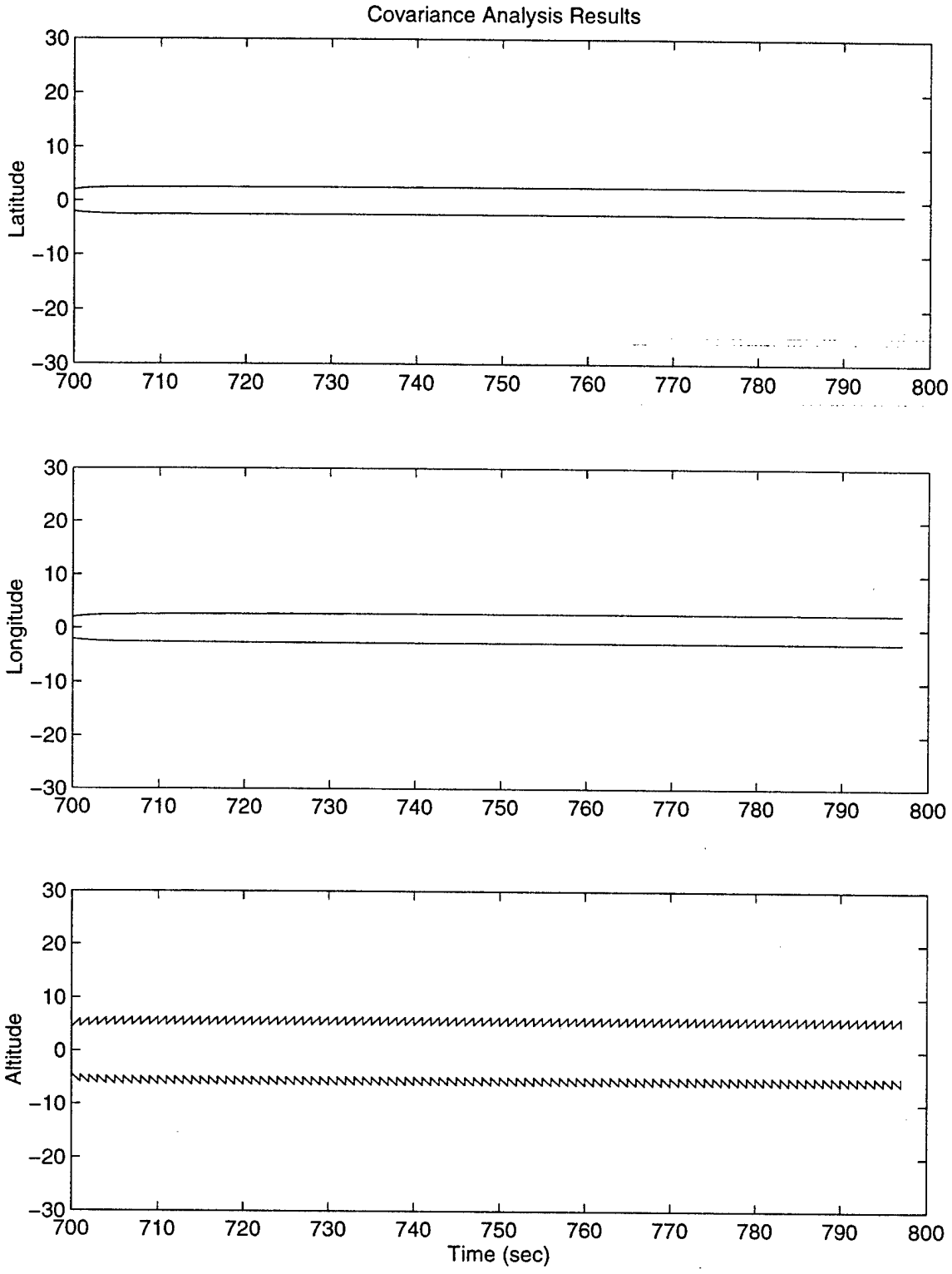


Figure 55. M^{3}AE Covariance Analysis: Test Case 1

bound on the M^{3}AE's real state estimation performance. This "over-approximation" is partly due to the "zeroth" order term of the covariance analysis tool (as given by Equation (108)) being too large by itself, and partly the addition of the first order term (as given by Equation (116)); due to any bias in the parameter estimate. Although it may seem strange that the "zeroth" order term is itself too large, since the M^{3}AE is using an EKF for the single state estimator rather than a linear KF, there is no proof that the filter-predicted error variance will match the true error variance (as *is* the case for a truth-model-based *linear* Kalman filter). This may also account for potential differences between the M^{3}AE approximate covariance analysis results versus actual performance. Thus for this example, the M^{3}AE approximate covariance analysis tool will most likely overestimate the M^{3}AE's actual performance in each test case.

Summary. Since all of the MMAEs tested in this case had an identical elemental filter, which received the majority of the probability weight throughout the simulation, the resulting state and parameter estimation performances were essentially identical. The resulting M^{3}AE performance is comparable to the MMAE performance, as anticipated.

4.2.3.2 Test Case 2: $a_T = 9.0, 6750.0, 9.0$ *Sequentially*

In this test case, the true parameter value undergoes a step change at the 718 second point in the simulation. It remains at the new true value, $a_T = 6750.0$ until the 760 second point in the simulation, upon which normal operation is returned. This sudden increase in the R_{GPS} value would simulate the instant "turn on" of a GPS signal interference source. Given MMAEs with only three elemental filters and the resulting coarse discretization, this value of $a_T = 6750.0$ is investigated since it provides a region in the parameter space where MMAE "blending" of the parameter estimates occurs with some consistency.

In addition to the analysis of the actual state and parameter estimation performance, the impact of Lund's IRDF is presented, followed by a comparison of the performance of the M^3AE's approximate covariance analysis versus actual M^3AE performance.

State Estimation Performance. Figures 56 and 57 show the plots from each MMAE's blended state estimation performance. Figure 58 shows the corresponding M^3AE state estimation performance. The temporally averaged RMS state estimation errors from each plot are summarized in Table 19 below.

Table 19. Test Case 2: Temporally Averged RMS State Estimation Errors (ft)

State	*MMAE/SDPEP* without IRDF	Conventional *MMAE/SDSEP*	M^3AE without IRDF
Lat	2.372	2.074	1.388
Long	3.108	2.742	2.144
Alt	12.32	12.1	8.337

In this test case, the figures and the table indicate that the *MMAE/SDSEP* has slightly better state estimation performance than the *MMAE/SDPEP.* The *MMAE/SDPEP* performs better than might be originally anticipated (given its parameter discretization values) due to the "good" blending of the estimates. The impact of this blending on the M^3AE state estimation performance is much more apparent. For example, notice the significant improvement in the altitude state (the bottom plot in the figures) in the M^3AE versus either MMAE. There are much smaller deviations in the mean error occurring during the interference period as compared to the same period in both of the MMAEs. Further notice that, during the remaining sections of the simulation (without real-world interference), performance is essentially the same for all the architectures, as anticipated given the results shown in test case 1.

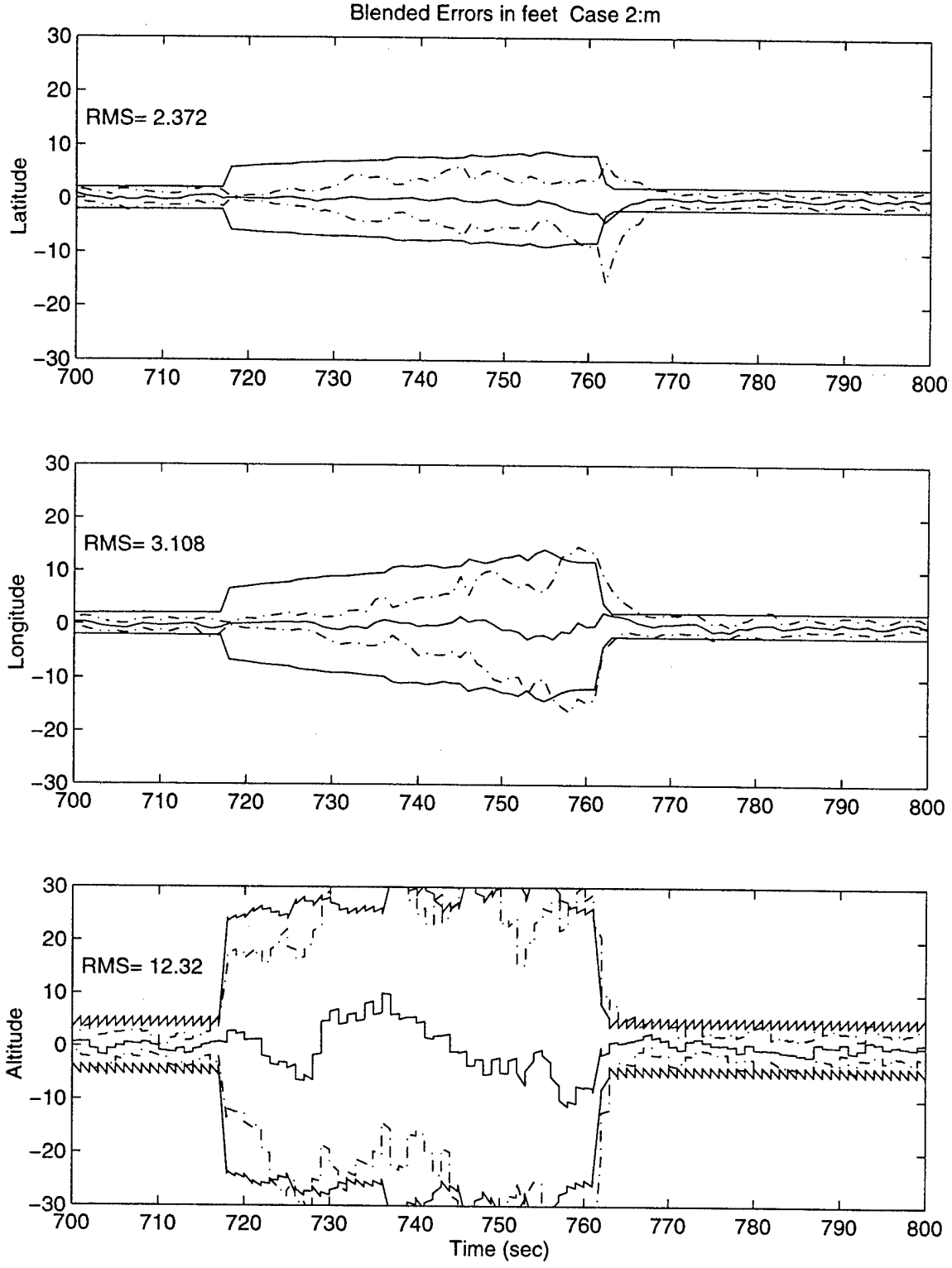


Figure 56. *MMAE/SDPEP* Without IRDF State Estimation Performance: Test Case 2

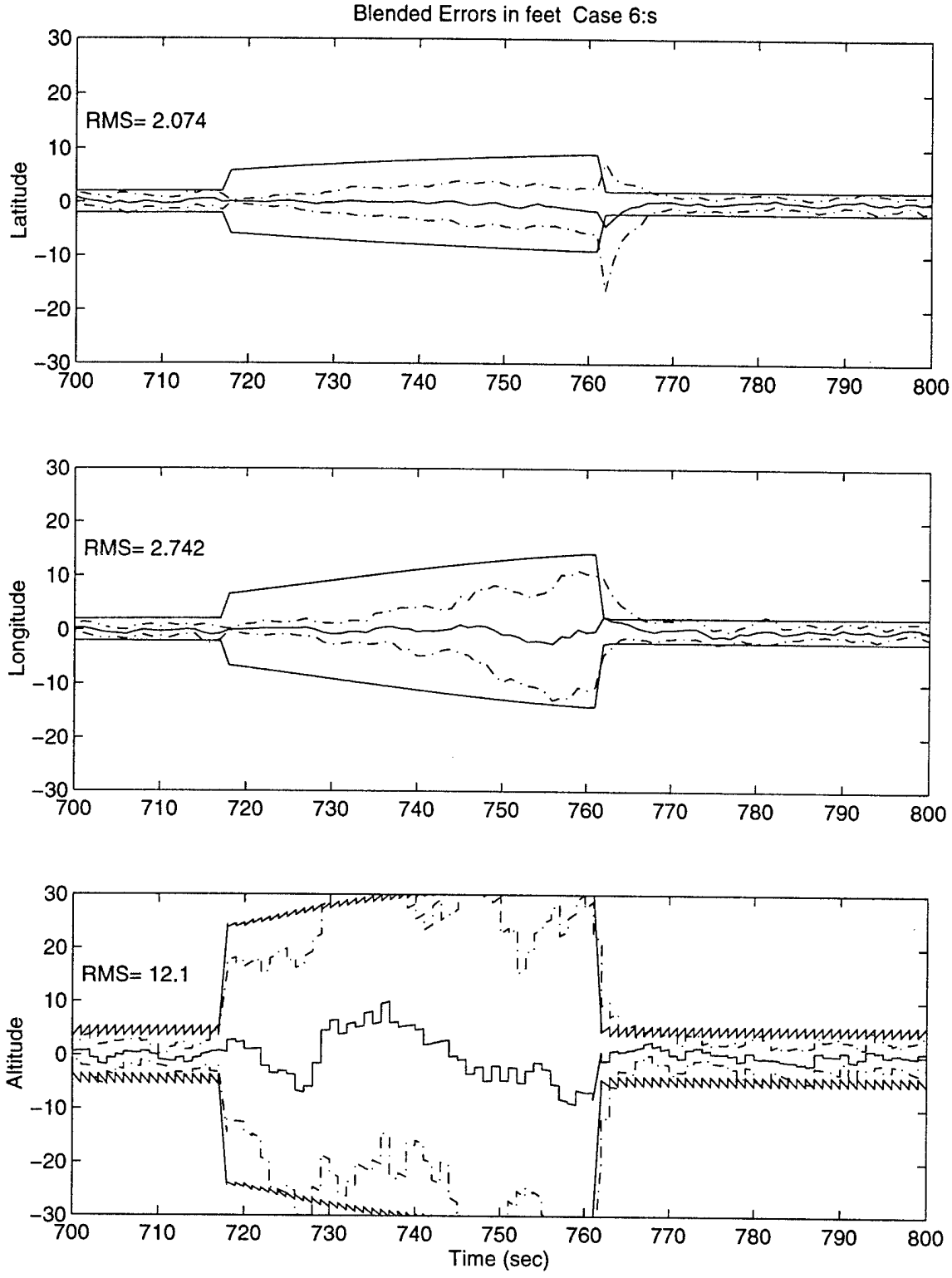


Figure 57. *MMAE/SDSEP* State Estimation Performance: Test Case 2

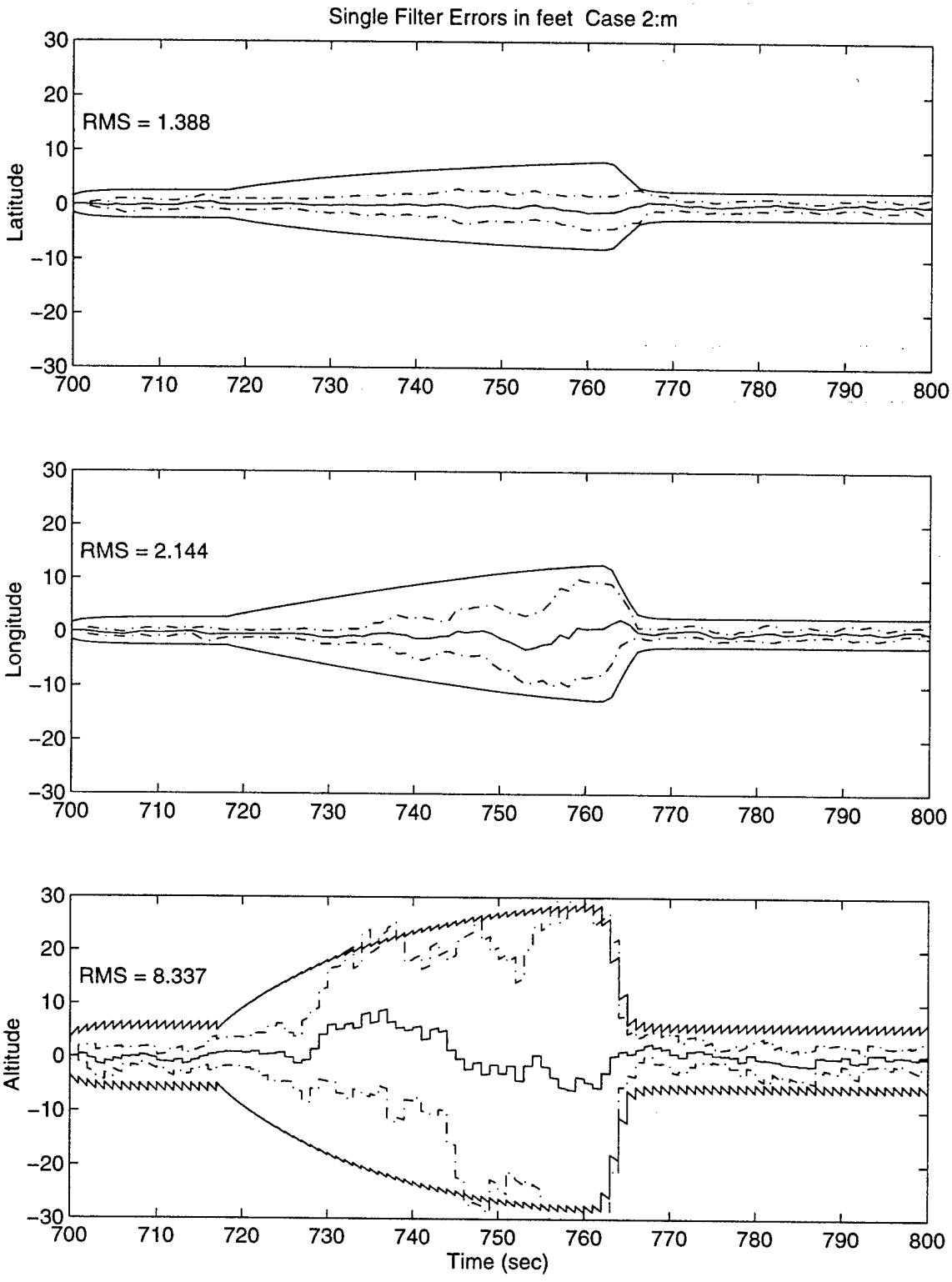


Figure 58. M^{3}AE Without IRDF State Estimation Performance: Test Case 2

The other significant distinction between performance in the M^3AE versus both of the MMAEs is the slow ramping of the filter-predicted standard deviation and true error mean $\pm$ one sigma performance curves in the M^3AE, versus the sharp rise in the corresponding curves in each of the MMAEs. This is again most evident in the altitude state. The sharp rise for the MMAEs is due to the "sudden switch" in the probability weighting to a different elemental filter (from elemental filter one to filter two, which more accurately represents this level of interference) in both MMAE filter banks. Despite the onset of interference, an accurate blended parameter estimate is provided to the M^3AE's state estimator, as evident by the near-zero-mean error and small mean $\pm$ one sigma performance. The M^3AE's state estimator does "react" to the interference as evident by the increase in the mean $\pm$ one sigma performance throughout the interference region. The slow rise is directly attributable to the standard propagate and update cycles in a Kalman filter. Recall, Equations (28) - (30), and (32) in Chapter 2; since $\mathbf{R}$ increases, the gain $\mathbf{K}$ decreases, and the benefit to the resulting update, $\mathbf{P}(t_i^+)$, value is smaller than for the previous update cycle. In turn, $\mathbf{P}(t_i^-)$ will grow until this transient period is over, thus the gradual increase in the slope of the covariance curves. Furthermore, notice that when the interference is removed at the 760 second point, all the architectures react and return to *normal* operation in approximately seven sample periods.

As a final indication of the M^3AE's state estimation performance for this test case, an additional 10-run Monte Carlo simulation is conducted, with $\hat{\mathbf{a}}$ from the internal MMAE replaced by $\mathbf{a}_T$ in the M^3AE's single Kalman filter. The test run will indicate the *best possible* performance that is achievable in this test case, given that the filter knows $\mathbf{a}_T$ exactly. Figure 59 plots the results. Notice that there is only a slight performance improvement over the M^3AE, given $\hat{\mathbf{a}}$, shown in Figure 58. This clearly highlights the potential benefit of the M^3AE architecture: it is very close to the ultimate bound of performance of a state estimator artificially given exact knowledge of the true parameter. Furthermore notice that, the filter-predicted σ performance overestimates the true error variance.

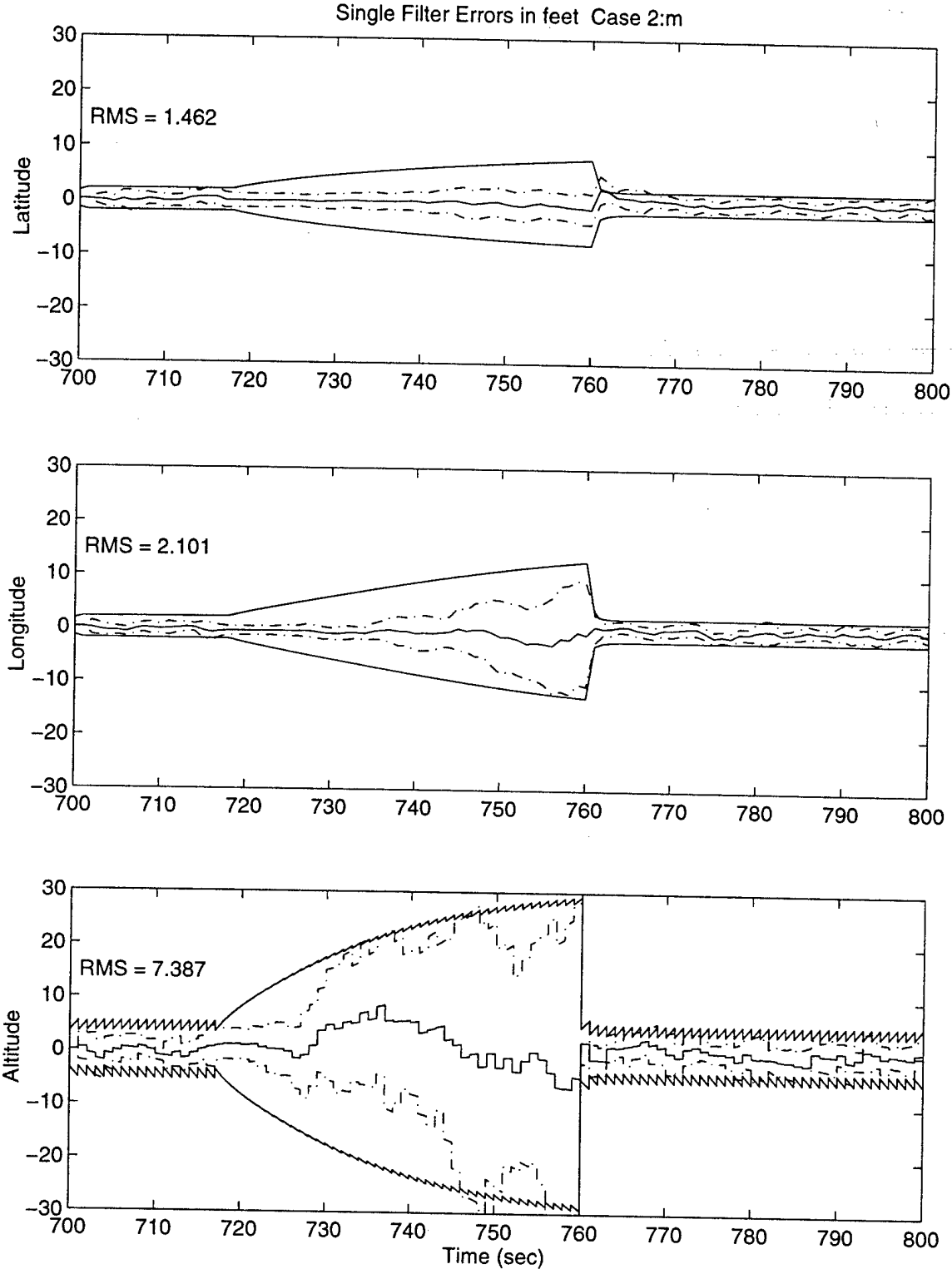


Figure 59. M^3AE State Estimation Performance Given a_T: Test Case 2

Again this is due to the fact that when using EKFs, there is no proof that the filter-predicted σ performance will match the true error variance performance (even if the true parameter values were known).

Parameter Estimation Performance. Figures 60 and 61 show the plots of each MMAE's blended parameter estimation performance. Notice in the bottom plot in Figure 60, the effect of blending on the *MMAE/SDPEP's* parameter estimation performance. During the region of the interference, a near-zero mean error is achieved. The exception occurs at the point where the interference is turned off. The sudden upward spike in the parameter estimate mean error, occurs since the majority of the probability weighting is given to the third elemental filter in the bank, which has the largest R_{GPS} value and subsequent smallest initial $[\mathbf{r}_j^{\mathrm{T}}(t_i)\mathbf{A}_j^{-1}(t_i)\mathbf{r}_j(t_i)]$ value, giving the bank the false impression that it has the best estimation performance. However, the *MMAE/SDPEP* quickly responds, once the residuals indicate that this performance is incorrect, and subsequent estimation performance improves.

Notice in Figure 61, that the *MMAE/SDSEP* has an undesirable bias present in the parameter estimate. This is a direct result of the *MMAE/SDSEP* placing the majority the probability weight on a single elemental filter, corresponding to $\mathrm{a}_2 = 8176$. Therefore, even though state estimation performance is acceptable, this biased parameter estimate is not.

Lund's Discrete-Time IRDF. Recall that IRDF's purpose is to increase the distinguishability between elemental filters in an MMAE bank. After several attempts at varying the tuning parameters, J_{jk}^0, $\eta_{\min}$, and ξ, it became apparent that IRDF provides little if any improvement in filter distinguishability and subsequent parameter estimation performance in this example problem. This is directly due to the nature of this problem which includes a coarse discretization over a large parameter space (thus distinguishability is inherent in the problem), an uncertainty in $\mathbf{R}$, and very small

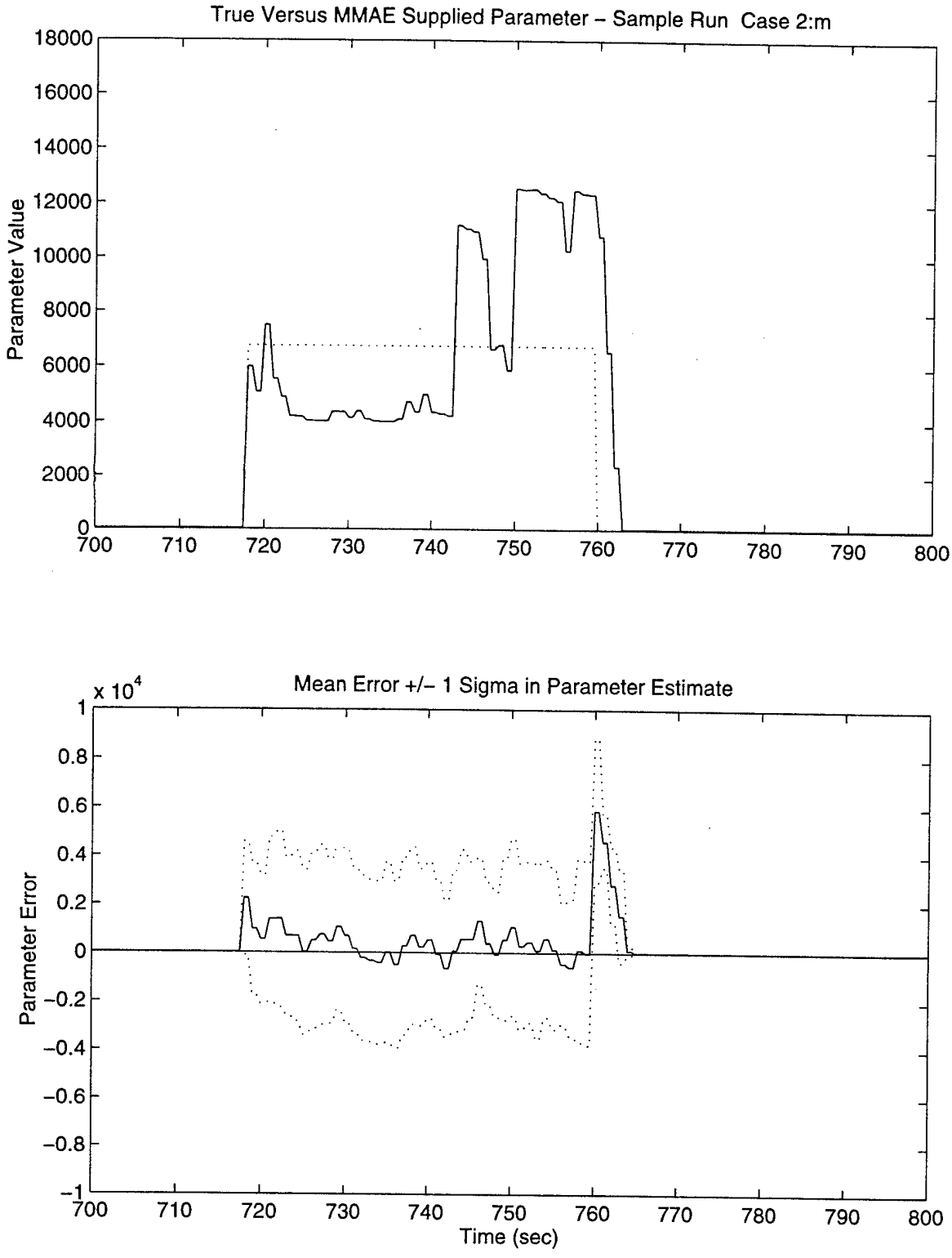


Figure 60. *MMAE/SDPEP* Without IRDF Parameter Estimation Performance: Test Case 2

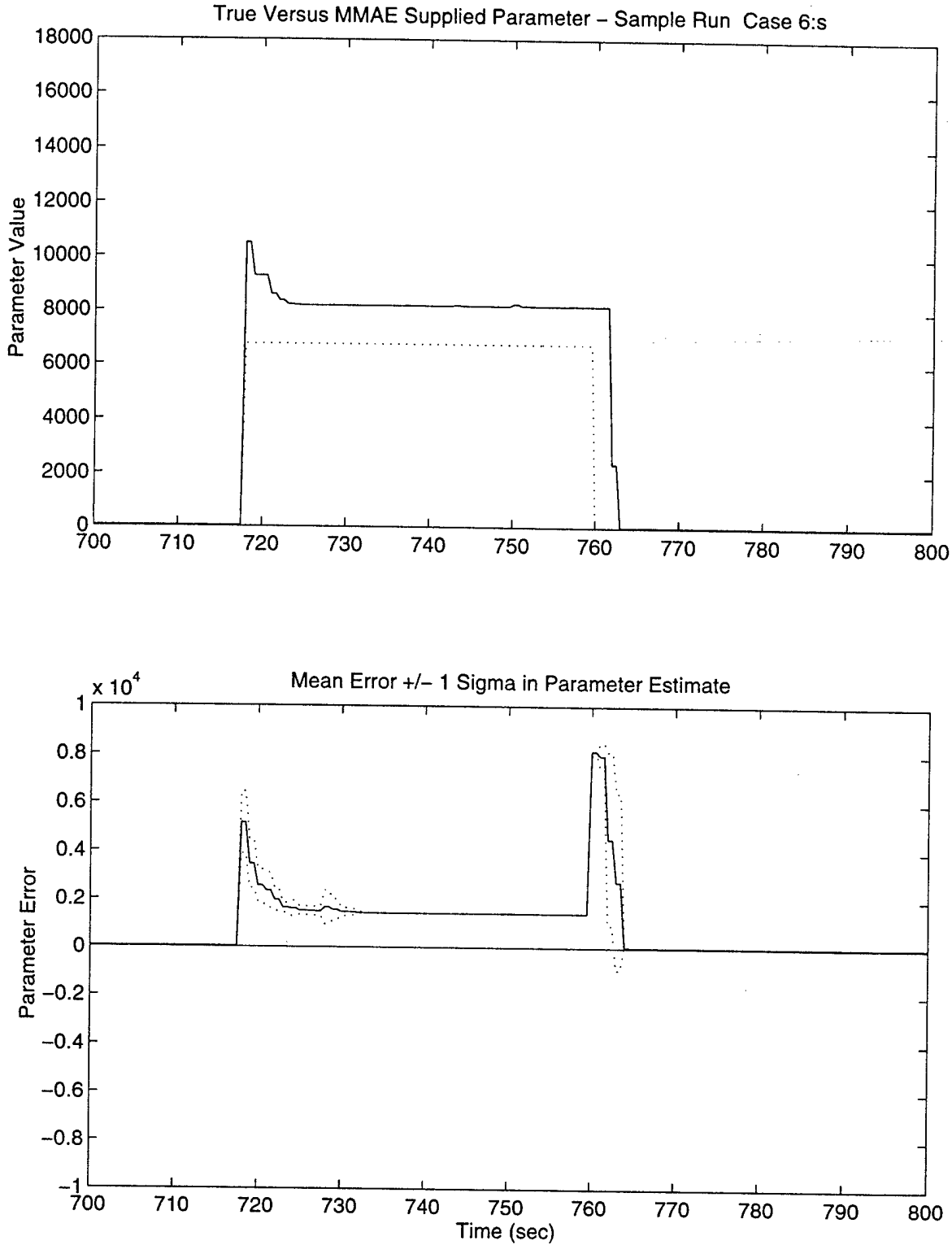


Figure 61. *MMAE/SDSEP* Parameter Estimation Performance: Test Case 2

Q_d values (very close to zero in most of the elements). Therefore, any modulation of the dynamics driving noise has little if any impact on parameter estimation performance.

This is evident upon inspection of Figures 62 - 64 and Table 20. The IRDF tuning parameters used in this test case example are: $J_{jk}^0 = 10,000, \eta_{\min} = 0.0$, and $\xi = 0.7$; however, note that all the tuning parameters investigated in this case ($[300 < J_{jk}^0 < 15,000]$, $[0 < \eta_{\min} < 0.5]$, and $[0.5 < \xi < 0.9]$) provide similar results. First notice that the MMAEs blended and subsequent M^{3}AE results are virtually the same. Thus, for this 13-state GPS/INS example problem, a comparison between the various architectures and IRDF is not accomplished in the other test cases.

Table 20. Test Case 2: Temporally Averged RMS State Estimation Errors (ft)

State	MMAE without IRDF	MMAE with IRDF	M^{3}AE without IRDF	M^{3}AE with IRDF
Lat	2.372	2.309	1.388	1.387
Long	3.108	2.99	2.144	2.139
Alt	12.32	12.22	8.337	8.331

M^{3}AE Approximate Covariance Analysis. Figure 65 shows the results of the approximate covariance analysis. Notice, in this test case, that the M^{3}AE approximate covariance analysis provides an accurate prediction of the actual M^{3}AE filter-computed σ performance of Figure 58 throughout the simulation (again, since the "zeroth-order" term, which is also given by the filter-computed σ performance of Figure 59, dominates the first order correction term for $\hat{a} \neq a_T$ but, by itself, is an overestimate of the *true* error standard deviation), but it is not as close to predicting the true mean $\pm 1\sigma$ performance. Additionally, in this example, the actual trajectory information related to the flight profile is not provided to the approximate covariance analysis tool for expediency. Instead, the time varying parameters related to the aircraft position are approximated by replacing them with the fixed values associated with the flight profile at the 750 second point (this in not inherent to the

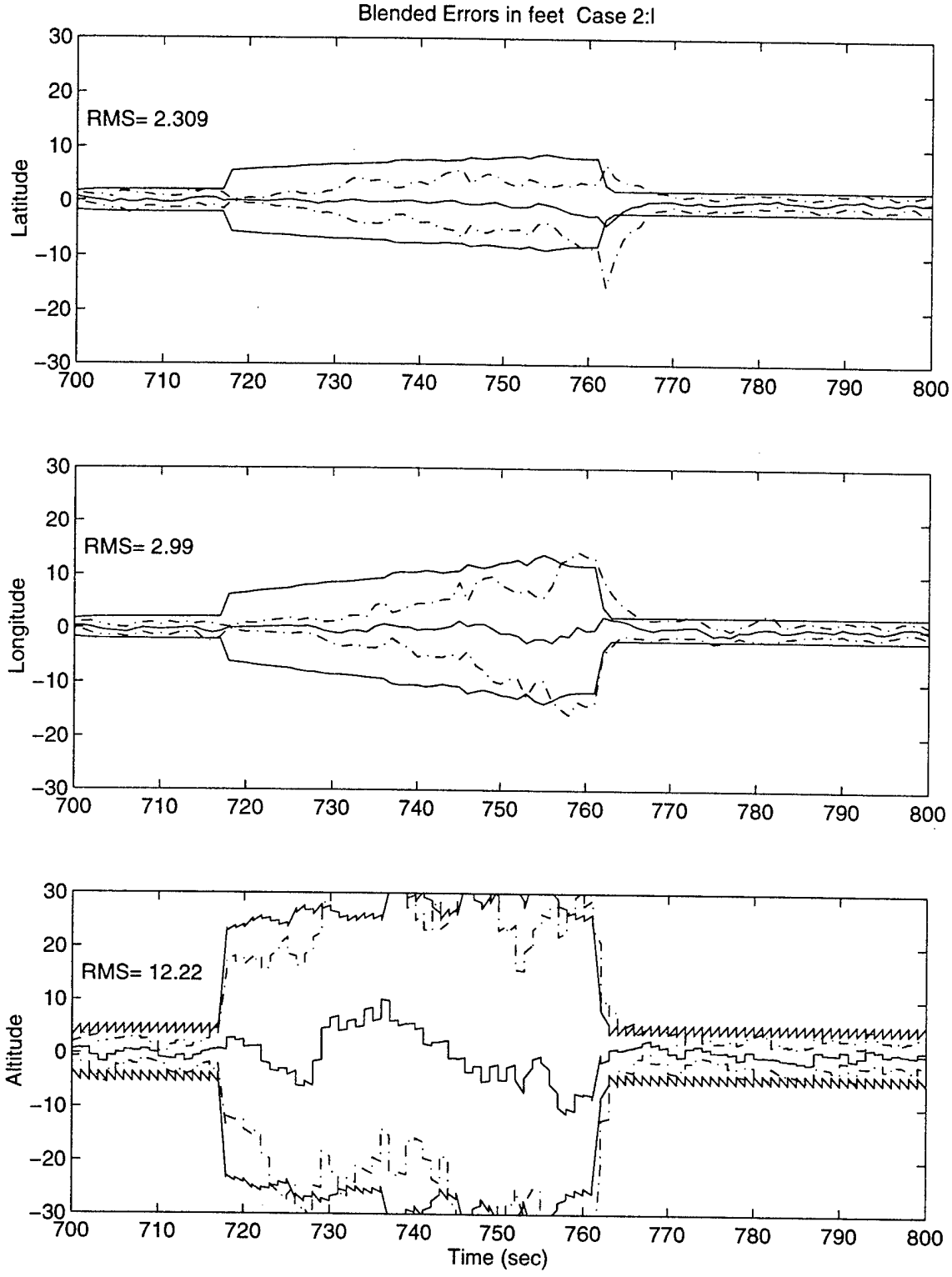


Figure 62. *MMAE/SDPEP* IRDF State Estimation Performance: Test Case 2

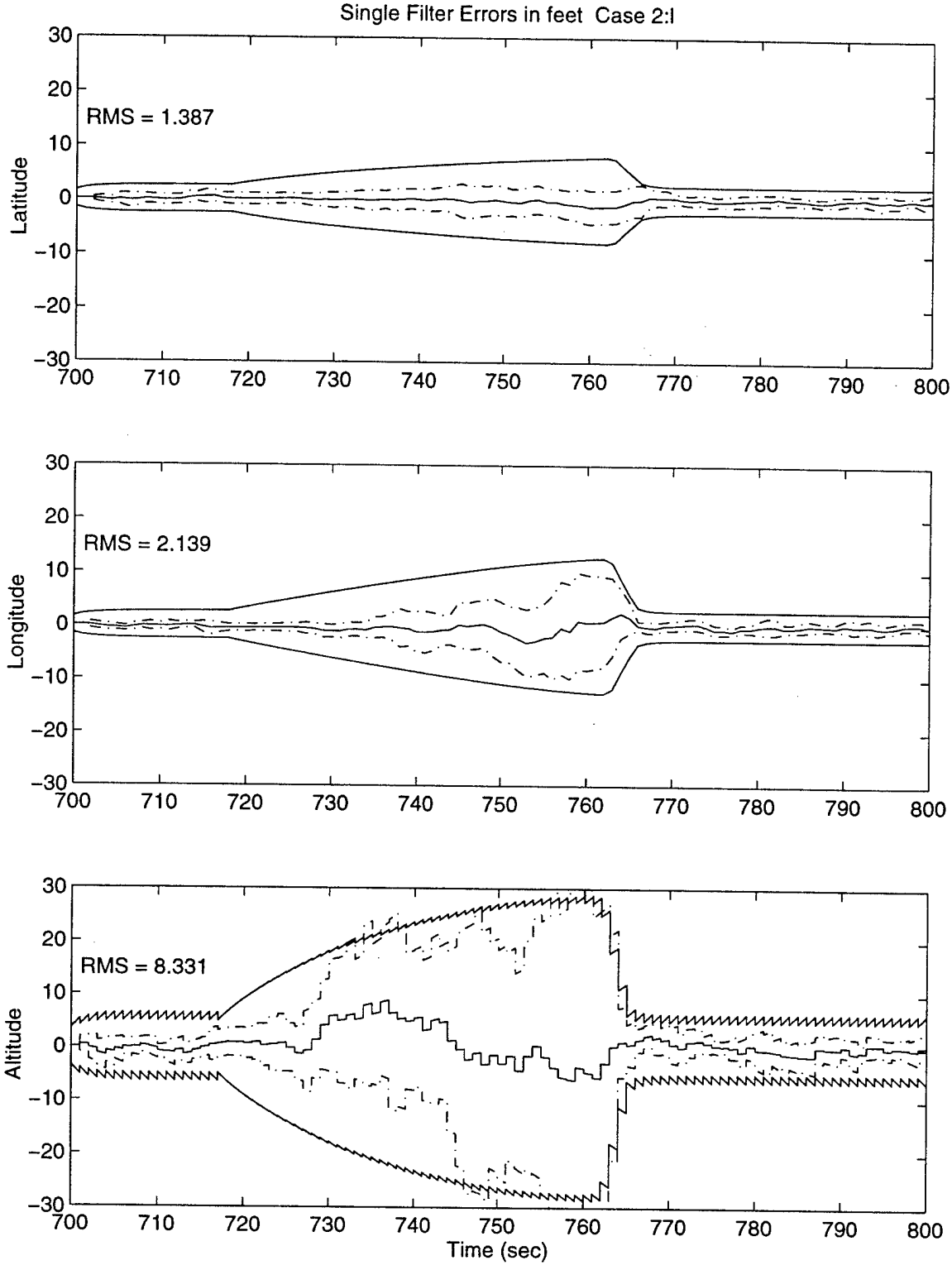


Figure 63. M^{3}AE With IRDF State Estimation Performance: Test Case 2

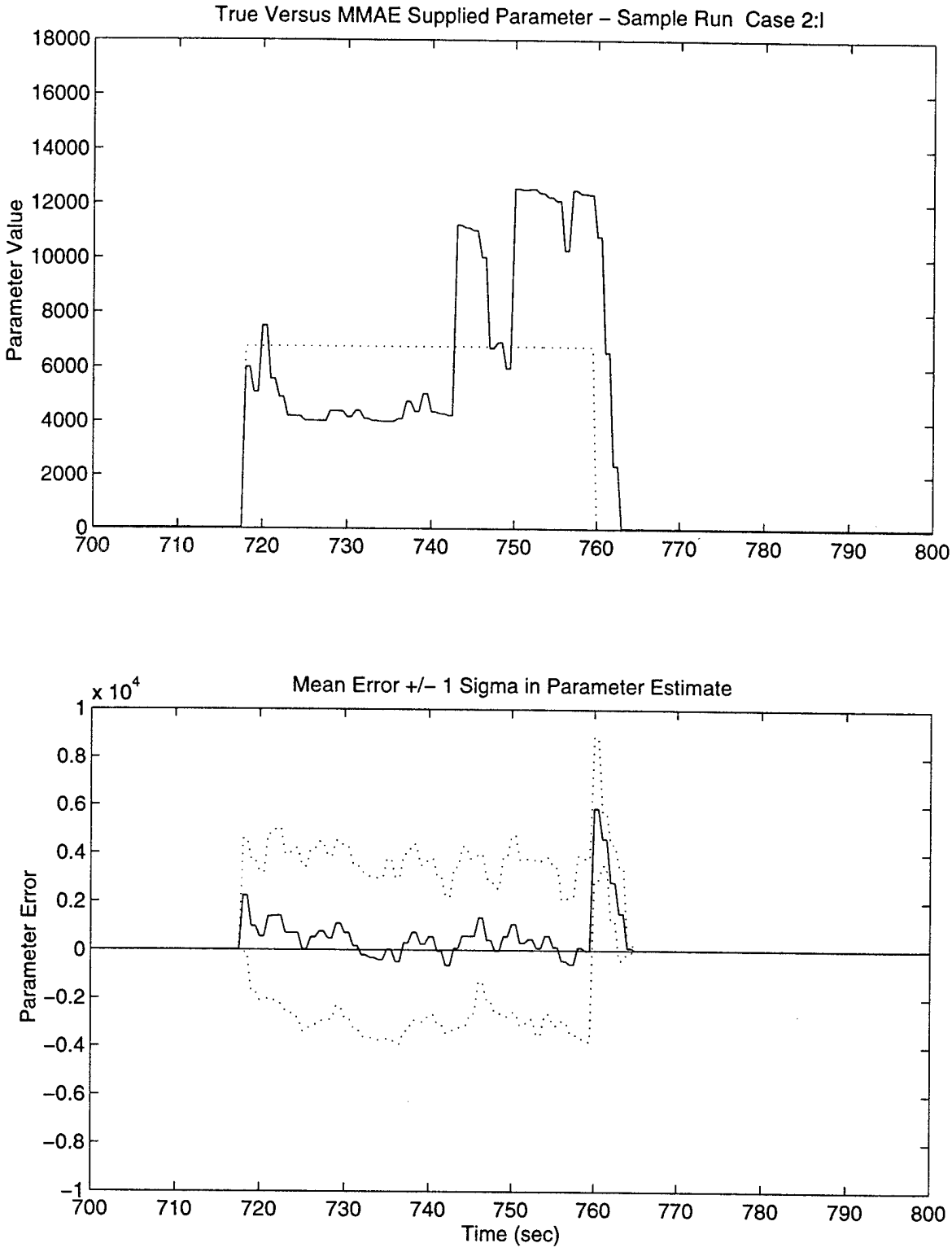


Figure 64. *MMAE/SDPEP* With IRDF Parameter Estimation Performance: Test Case 2

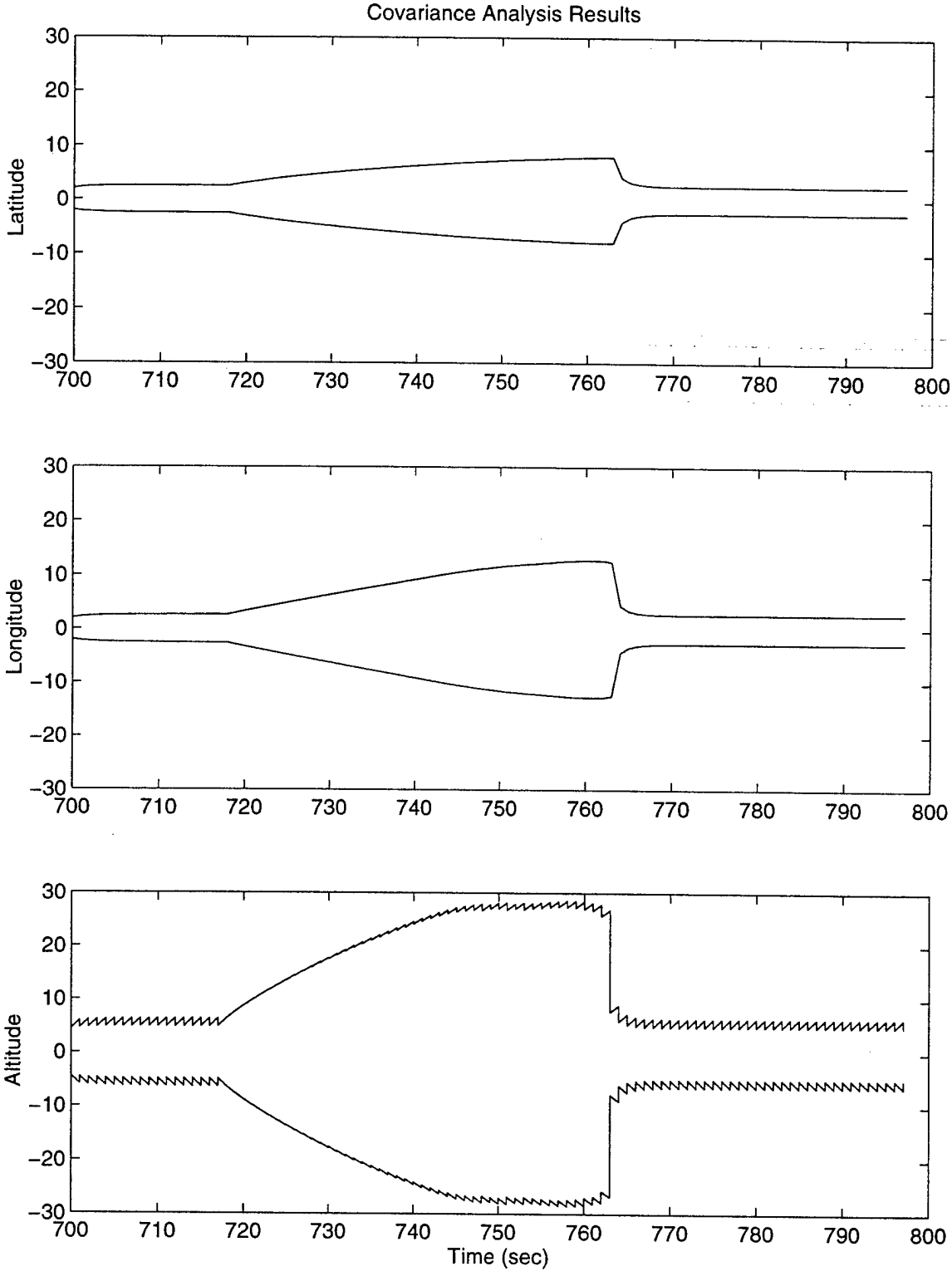


Figure 65. M^{3}AE Covariance Analysis: Test Case 2

use of the tool and was not discussed in Chapter 3, or really necessary to the use of the tool). A more accurate reflection of the covariance performance is achievable by incorporating the trajectory information into the approximate M^{3}AE covariance analysis tool (this could become very important, especially when longer flight profiles and more dynamic maneuvers are simulated). However, even given this flight profile restriction, the covariance analysis results still reflect a good upper bound indication of expected performance for this example. Additionally, as discussed in the previous test case, the use of EKFs and the effects caused by the "zeroth-order" term of the approximate covariance being itself too large, tend to prevent an accurate prediction of the true mean $\pm 1\sigma$ performance.

Summary. This test case highlights the performance benefits associated with the M^{3}AE architecture. Given good *MMAE/SDPEP* blending to produce near-zero-mean-error parameter estimates, the M^{3}AE state estimation performance should be better that the associated *MMAE/SDSEP* state estimation performance. Moreover, it is very close to the ideal performance bound on state estimation, as though parameter estimation errors were perfectly zeroed out.

4.2.3.3 Test Case 3: $a_T = 9.0, 9000.0, 4500.0, 9.0$ *Sequentially*

In this test case, the true parameter value undergoes three step changes at the 700, 718, and 760 second points in the simulation. It initially starts the simulation at the parameter value $a_T = 9.0$, then at $t_i = 700$ seconds, a_T undergoes a step change to a high level of interference equal to 9000.0, at which point the GPS measurements are assumed to be "poor" in this test case. This test case demonstrates the impact on the M^{3}AE's performance when *MMAE/SDPEP* accurate blending is not present during the majority of the simulation. Again analysis of the actual state and parameter estimation performance is presented, followed by a comparison of the performance of the M^{3}AE's approximate covariance analysis versus actual M^{3}AE performance.*State Estimation Performance.* Figures 66 and 67 show the plots from each MMAE's blended state estimation performance. Figure

68 shows the M^{3}AE state estimation performance. The temporally averaged RMS state estimation errors from each plot are summarized in Table 21 below.

Table 21. Test Case 3: Temporally Averged RMS State Estimation Errors (ft)

State	*MMAE/SDPEP* without IRDF	Conventional *MMAE/SDSEP*	M^{3}AE without IRDF
Lat	2.442	2.052	1.844
Long	3.379	2.854	3.098
Alt	13.04	13.19	12.87

In this test case, the figures and the table indicate that state estimation performance in both the MMAEs is similar. There is a small performance benefit with the M^{3}AE architecture is this test case also, but that is due to the problem setup. The *MMAE/SDPEP* 's elemental filters $\mathbf{a}_j$'s are *closer* to $\mathbf{a}_T$ for longer periods of time than the *MMAE/SDSEP* 's elemental filters $\mathbf{a}_j$'s (recall test case 8 in Section 4.1). Good blending (indicated by the parameter estimate having a zero-mean bias as in test case 2) is absent during the majority of the simulation; therefore, the M^{3}AE's state estimation performance does not significantly improve over that of the MMAE, as in test case 2 above. Furthermore note that, even though the M^{3}AE does not produce a significant increase in state and parameter estimation performance, it doesn't deteriorate performance either. Finally, notice that the initial transients shown in plots are due to the fact that the initial parameter value was at $\mathbf{a}_T = 9.0$, then at $t_i = 700$ seconds, $\mathbf{a}_T$ undergoes a step change to 9000.0. Again the slow rise in the mean $\pm 1\sigma$ performance is directly attributable to the standard propagate and update cycles in elemental filters. Finally notice that the effects of the step change at $t_i = 718$ seconds are not visible, since the mean $\pm 1\sigma$ performance is still *growing towards* steady state values from the initial step change from 9.0 to 9000.0 at $t_i = 700$ seconds.

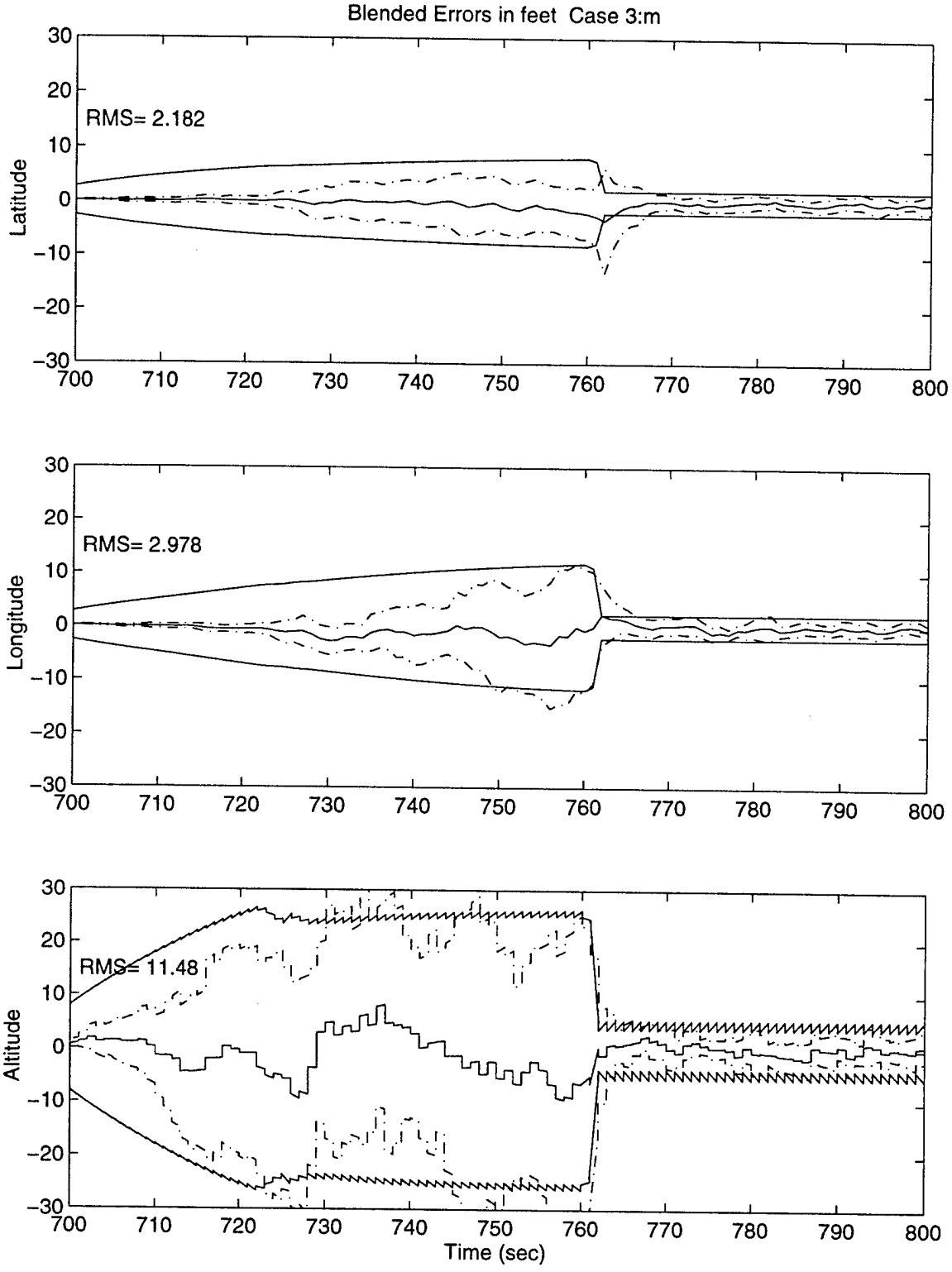


Figure 66. *MMAE/SDPEP* Without IRDF State Estimation Performance: Test Case 3

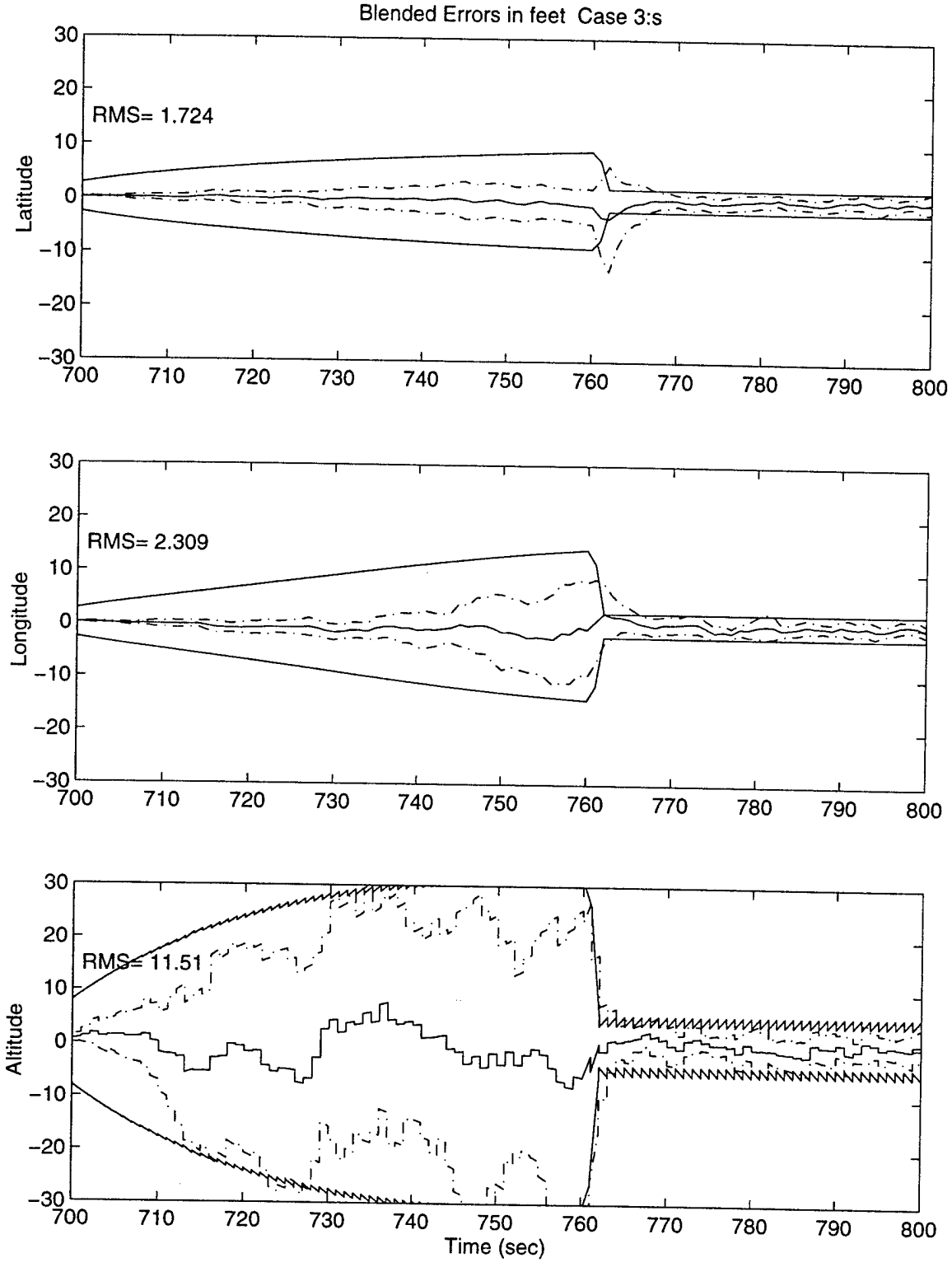


Figure 67. *MMAE/SDSEP* State Estimation Performance: Test Case 8

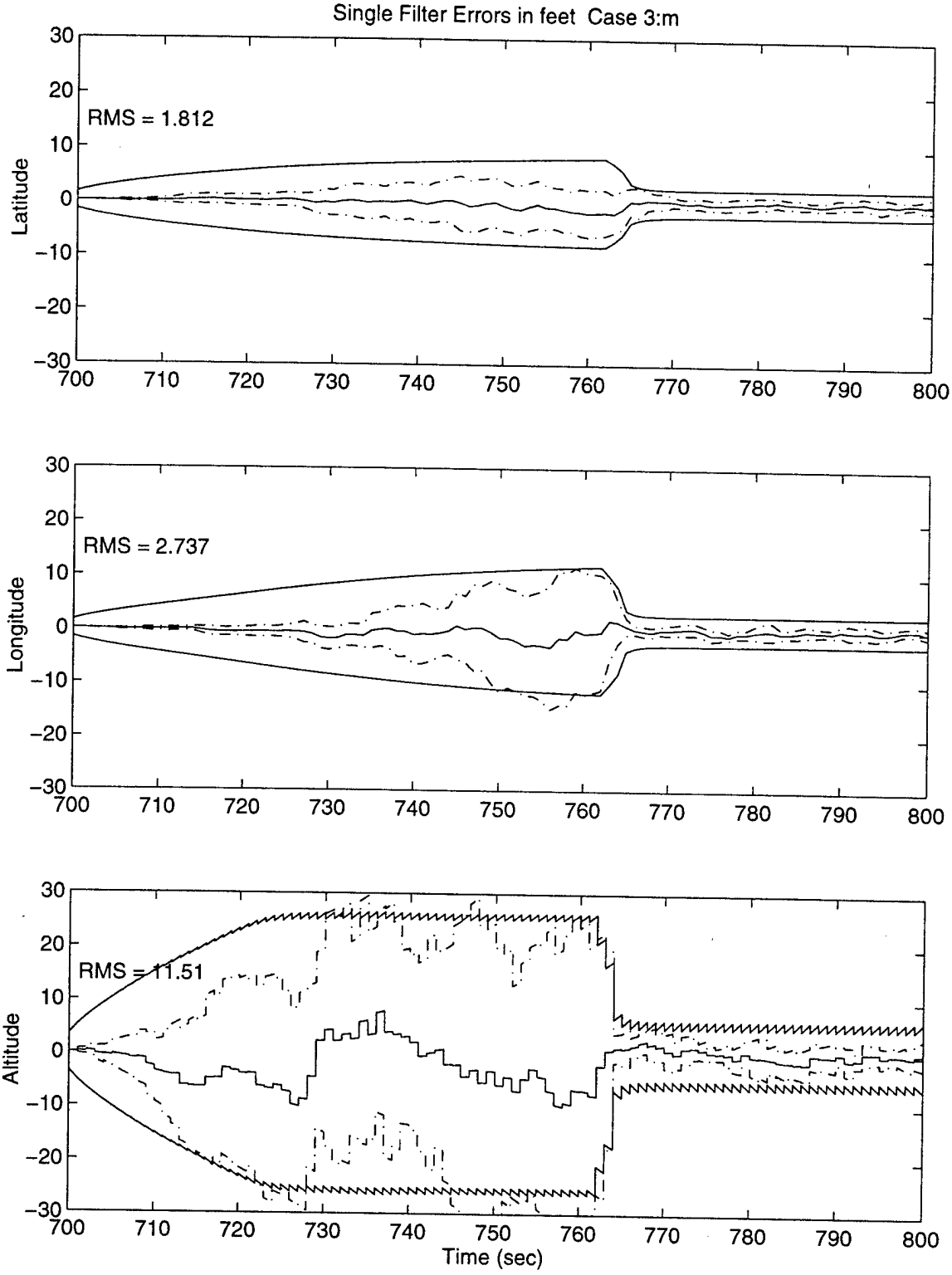


Figure 68. M^{3}AE Without IRDF State Estimation Performance: Test Case 8

Parameter Estimation Performance. Figures 69 and 70 show the plots of each MMAE's blended parameter estimation performance. Notice in the bottom plot in Figure 69, that "blending" of the parameter estimates is occurring; however, there is a distinct bias in all but 10 seconds of the simulation. Furthermore, notice in the top plot of Figure 69 that, for the first 18 seconds, the parameter estimate is approximately equal to $a_2 = 3998.0$, which is due to the fact that the true parameter value was at $a_T = 9.0$, up until $t_i = 700$ seconds. Then, once the initial transient has passed, the *MMAE/SDPEP* switches to the elemental filter *closest* to the $a_T = 9000.0$, which is $a_3 = 12,578.0$. Additionally, the parameter estimate in Figure 70 reflects the parameter value of the elemental filter *closest* in the "Baram distance measure sense" to the *MMAE/SDSEP* second elemental filter, $a_2 = 8176.0$. The state and parameter estimation in this test case suffers as a direct result of the coarse discretization of the parameter space.

M^{3}AE Approximate Covariance Analysis. Figure 71 shows the results of the approximate covariance analysis. As in test case 2, the M^{3}AE approximate covariance analysis provides a good indication of the filter-computed σ plots and subsequent upper bound on the expected M^{3}AE state estimation performance throughout the simulation.

Summary. This test case indicates that the M^{3}AE performs just as well in state estimation performance as the *MMAE/SDSEP*, despite not having the benefit of accurate parameter estimates. Since effective blending does not take place most of the time in this case, the M^{3}AE does not outperform the *MMAE/SDSEP*, as it did in the preceding case.

4.3 Summary

The M^{3}AE architecture demonstrates significant performance potential for estimating both parameters and states simultaneously. In most cases the M^{3}AE outperforms the conventional MMAE tuned and discretized for state estimation. The exceptions occur when the true parameter value is

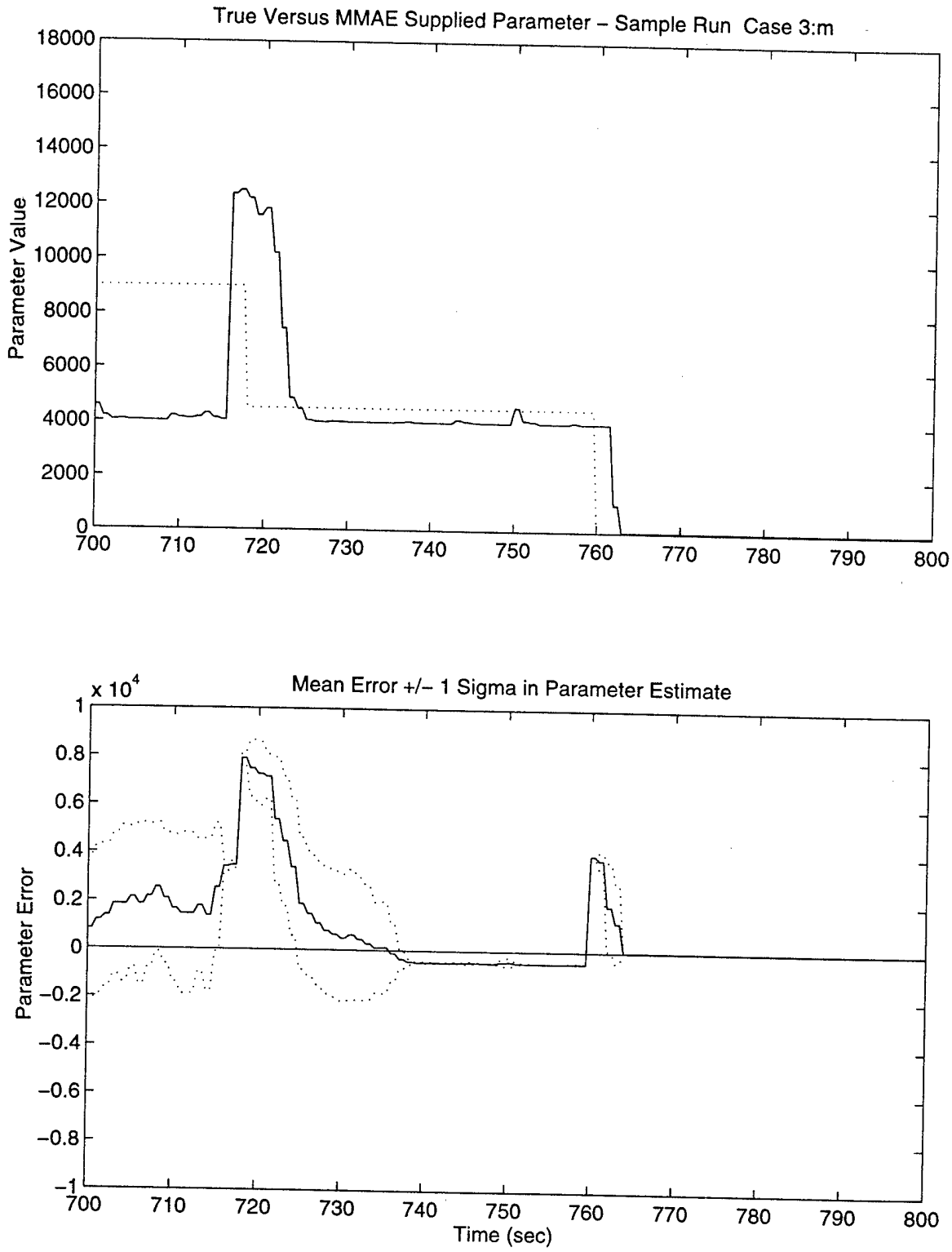


Figure 69. *MMAE/SDPEP* Without IRDF Parameter Estimation Performance: Test Case 8

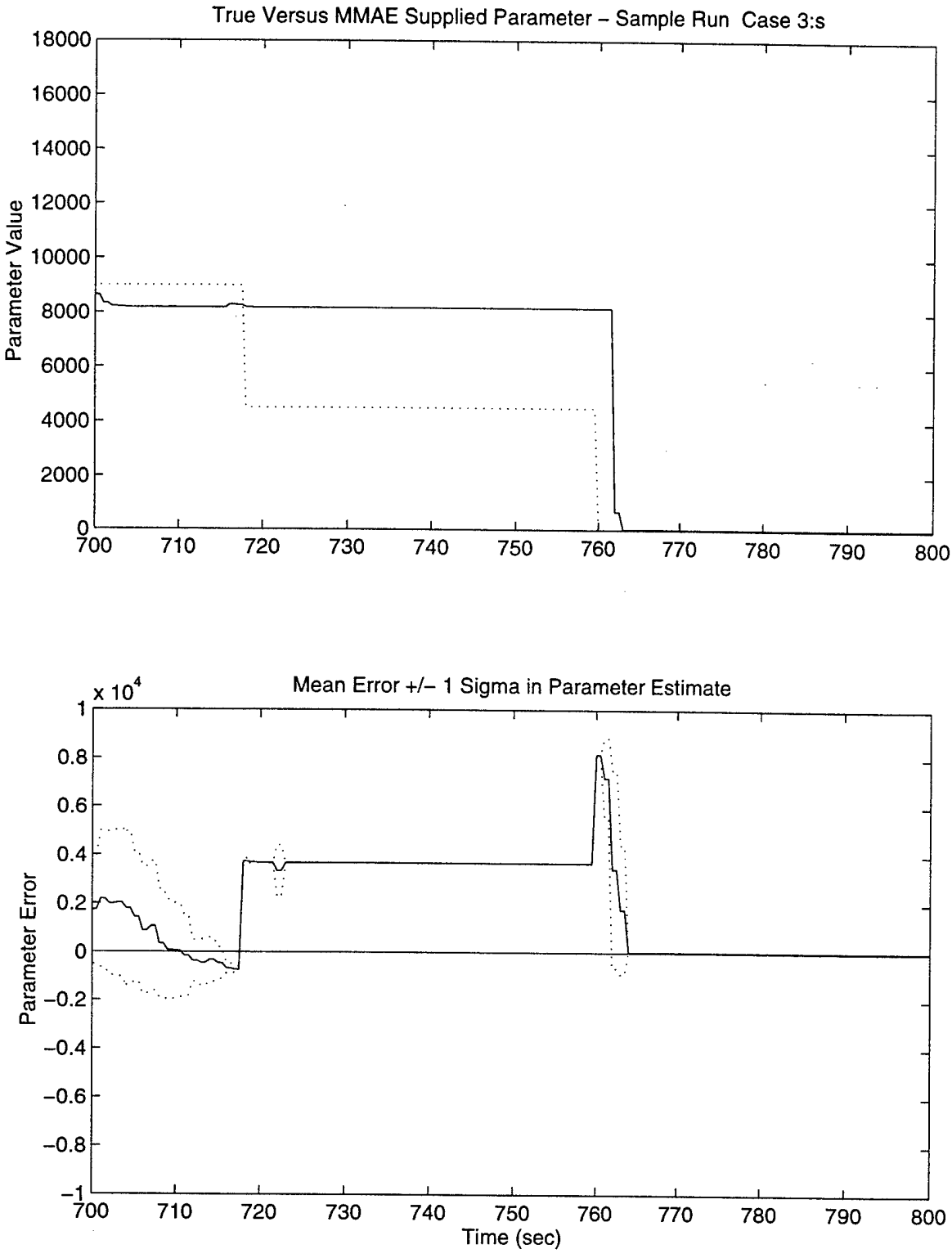


Figure 70. *MMAE/SDSEP* Parameter Estimation Performance: Test Case 8

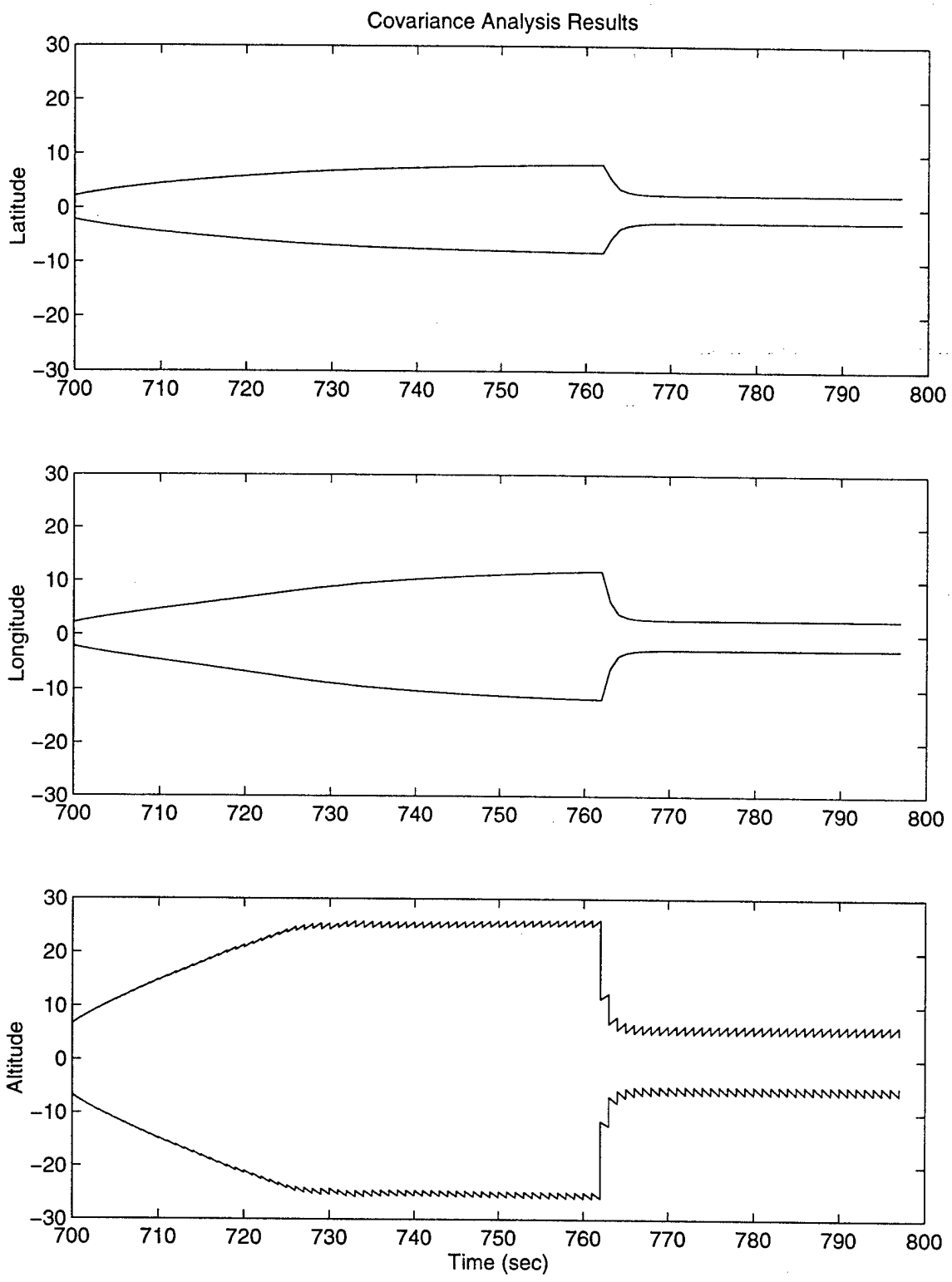


Figure 71. M^{3}AE Covariance Analysis: Test Case 3

closer to a parameter value used for the basis of an elemental filter in the conventional MMAE than to any discrete parameters in the elemental filters in the M^3AE, resulting in a larger bias in the M^3AE's parameter estimate. This yields a single state estimator based on an incorrect parameter value and thus results in degraded performance. The M^3AE results indicate that the better the parameter estimate, the better the state estimation performance. This is accomplished when there is effective blending in the *MMAE/SDPEP* within the M^3AE, providing an $\hat{\mathbf{a}}$ substantially closer to the true $\mathbf{a}_T$ value than any of the discrete $\mathbf{a}_j$ values used as the basis for that MMAE's elemental filters. Thus, accounting for or minimizing the parameter estimation bias with a consistently "well" blended parameter estimate is the next logical step required to improve the overall M^3AE's state and parameter estimation performance. The next chapter will address an effective means of ensuring that such blending does take place consistently.

Chapter 5 - Conclusions and Recommendations

5.1 Conclusions

A new contribution has been made in solving the problem of estimating states and parameters simultaneously, for linear, dynamic, sampled-data systems. The M^3AE architecture is one solution to eliminating the typical trade-off decision faced by engineers in developing designs intended for accurate state estimation versus designs intended for accurate parameter estimation.

The M^3AE architecture uses a combination of existing multiple model and Kalman filter-based methods to provide the accurate state and parameter estimation given parameter variations in any of the Φ, $\mathbf{B}_d$, $\mathbf{G}_d$, $\mathbf{H}$, $\mathbf{Q}_d$, and $\mathbf{R}$ system matrices. The M^3AE exploits the benefits of an MMAE designed for accurate parameter estimation, and yet performs at least as well in state estimation as an MMAE designed for accurate state estimation. The M^3AE accomplishes the simultaneous estimation task by providing accurate state estimates from a single KF designed to accept accurate parameter estimates from the MMAE within its architecture.

To assist the engineer in designing an M^3AE for a given sampled-data problem, a seven-step technique was developed. This technique gives the designer the ability to analyze, tune, and predict system performance before the M^3AE is constructed and subjected to a full-scale Monte Carlo analysis. Several contributions were made during the development of this algorithm.

1. Sheldon's five-step algorithm [69, 70] for determining the *best* parameter discretization for an MMAE was enhanced to account for linearized (versus purely linear), unstable or astable systems which may never attain steady state. The extension involves using a finite horizon assumption and a constrained-range optimization to solve the minimization in order to determine the *best* parameter discretization for a given problem.

2. Lund's IRDF technique [34,35], which increases the distinguishability between elemental filters in an MMAE bank, was extended for discrete-time systems, resulting in enhanced M^{3}AE parameter estimation.

3. Finally, the most significant contribution was the development of an approximate M^{3}AE covariance analysis design tool, based on a first order approximation to the error in the M^{3}AE's state estimate. The approximate M^{3}AE covariance analysis design tool assists the designer in predicting M^{3}AE performance after conducting only a single Monte Carlo simulation on the MMAE-based parameter estimator. The approximate M^{3}AE covariance is itself sometimes approximated by its upper bound, the corresponding approximate error correlation, because of the computational savings of so doing. However, the accuracy of the results is very dependent upon the accuracy of the MMAE-supplied parameter estimate. A large bias in the parameter estimate causes the approximate M^{3}AE covariance analysis design tool to produce a larger-than-expected prediction of covariance performance. If it is desired to account for the bias, the "square of the mean" term, $E\{\mathbf{e}_{M^3AE}(t_i)|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\}E\{\mathbf{e}_{M^3AE}(t_i)^\mathrm{T}|\mathbf{a}_\mathrm{T},\hat{\mathbf{a}}\}$, may be subtracted from the approximate error correlation matrix, $\mathbf{\Psi}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_\mathrm{T},\hat{\mathbf{a}})$, to form the *more accurate* approximate error covariance, $\mathbf{P}_{\mathbf{e}_{M^3AE}}(t_i;\mathbf{a}_\mathrm{T},\hat{\mathbf{a}})$.

To verify this new architecture's expected performance, the M^{3}AE architecture, algorithm, and approximate covariance analysis tool were validated against two examples. The results demonstrated the application of the theory and performance achieved with an M^{3}AE compared to conventional MMAEs. The M^{3}AE, in general, outperformed the conventional MMAE tuned and discretized for state estimation. The most significant improvement in state estimation performance occurred between the M^{3}AE/IRDF versus the *MMAE/SDPEP/*IRDF. The improvement occurs since a blended "near-zero-mean error" parameter estimate is provided to the M^{3}AE's state estimator. Fur-

thermore, when blending does not occur, the subsequent *MMAE/SDPEP* (with or without IRDF) state estimation performance is similar to the M^3AE's state estimation performance, since both have biased estimates. The M^3AE's state estimation performance also suffered when the true parameter value was *closer* to a discrete parameter value used for the basis of an elemental filter in the conventional MMAE bank (discretized for state estimation performance) than to the discrete values in any of the elemental filters in the M^3AE's MMAE bank (discretized differently for best possible parameter estimation performance), resulting in a larger bias in the M^3AE's parameter estimate. The resulting state estimation performance suffered since the state estimates were based on a "biased" parameter estimate. Accounting for the bias estimates with effective MMAE blending is the next logical step in increasing the M^3AE overall performance in state and parameter estimation.

Moreover, the cases in which the M^3AE performed most significantly better than the MMAE were the cases in which blending of two or more elemental filter outputs occurred, rather than one elemental filter receiving essentially all the probability weight. Thus, the $\hat{\mathbf{a}}$ from the MMAE within the M^3AE algorithm would be substantially different from (and better than) any of the hypothesized $\mathbf{a}_j$ values used as a basis for that MMAE's elemental filters. To produce such blending of multiple elemental filters in practice, the discretization of the parameter space must be fine enough that a single elemental filter does not typically absorb essentially all of the probability weight. (If any elemental filter does absorb essentially all the probability weight, then the M^3AE's single filter will be virtually indistinguishable from that elemental filter and the corresponding MMAE in performance). Such a need for fine discretization (while not requiring and inordinate number of elemental filters to cover the entire admissible parameter space), along with the previously mentioned need to reduce the bias in the parameter estimation, argue strongly for M^3AE to be applied to moving-bank versus fixed-bank, multiple model adaptation.

5.2 Recommendations

The M^{3}AE architecture developed in this research is a viable solution to the problem of estimating states and parameters simultaneously. However, there are certain areas that warrant further research.

1. The M^{3}AE provides the greatest improvement in state and parameter estimation when filter blending occurs. M^{3}AE results indicate that, the better the parameter estimate, the better the state estimation performance. Thus, there is strong motivation to try to reduce any bias in the parameter estimate. The first alternative is to implement an MMAE parameter estimator bank with a finer discretization requiring more elemental filters. However, this could quickly become an unreasonable computational burden. Therefore, replacing the M^{3}AE's fixed-bank MMAE with a moving-bank MMAE should further enhance parameter and subsequent state estimation performance.

2. The bias term, $E\{\mathbf{e}_{M^3AE}(t_i)|\mathbf{a}_T, \hat{\mathbf{a}}\}E\{\mathbf{e}_{M^3AE}(t_i)^T|\mathbf{a}_T, \hat{\mathbf{a}}\}$, was ignored in the approximate M^{3}AE covariance analysis tool for expediency and the desire to determine only an *upper bound* on the approximate covariance. However, accounting for this term may assist in some performance evaluations. Therefore, investigations into the impact of augmenting the M^{3}AE approximate covariance analysis tool with the bias term is recommended. Accounting for the bias term should provide a better approximation to the *total* covariance and, in turn, provide a better indication of the achievable M^{3}AE state estimation accuracy, especially for the worst case scenarios in which the true parameter value, $\mathbf{a}_T$, is *far* from any of the MMAE elemental filter's assumed $\mathbf{a}_j$'s in the current discretization.

3. The examples studied in this research investigated the M^{3}AE's performance with only one parameter value changing in time. Cases with more than one unknown parameter value should

also be investigated to verify the M^3AE's ability to handle multiple parameters effectively.

4. The M^3AE architecture is a strong candidate for implementation in estimation problems. The next logical extension is to apply the M^3AE architecture to control problems involving trackers, regulators, proportional plus integral (PI) controllers, or other control techniques. In addition to applying the M^3AE architecture to a control problem, investigation into parameter uncertainties involving the $\mathbf{B}_d$ system matrix has yet to be accomplished and is highly recommended.

APPENDIX A - Model State Definitions and System Matrices

This appendix contains a tabular listing of the 13-state GPS/INS reduced-order model. The LN-93 error-state dynamics matrix **F** and the process noise matrix **Q** as provided by Litton are 93-by-93 arrays containing a large number of elements that are identically zero [25]. The *non-zero* elements of the Litton model that apply to the reduced-order 13-state model are included in Tables 22 through 27.

Table 22. Reduced-Order System Model States

State Number	State Symbol	Definition	LN-93 State	PLS State
1	$\delta\theta_x$	X-component of vector angle from true to computer frame	1	1
2	$\delta\theta_y$	Y-component of vector angle from true to computer frame	2	2
3	$\delta\theta_z$	Z-component of vector angle from true to computer frame	3	3
4	ϕ_x	X-component of vector angle from true to platform frame	4	4
5	ϕ_y	Y-component of vector angle from true to platform frame	5	5
6	ϕ_z	Z-component of vector angle from true to platform frame	6	6
7	δV_x	X-component of error in computed velocity	7	7
8	δV_y	Y-component of error in computed velocity	8	8
9	δV_z	Z-component of error in computed velocity	9	9
10	δh	Error in vehicle altitude above reference ellipsoid	10	10
11	δh_B	Total baro-altimeter correlated error	23	11
12	δPR_{Uclk}	GPS User clock bias	-	12
13	δD_{Uclk}	GPS User clock drift	-	13

Table 23. Elements of the Dynamics Submatrix $\mathbf{F}_{(red)11}$

Element	Term	Element	Term
(1,3)	$-\rho_y$	(1,8)	$-C_{RY}$
(2,3)	ρ_x	(2,7)	C_{RX}
(3,1)	ρ_y	(3,2)	$-\rho_x$
(4,2)	$-\Omega_z$	(4,3)	Ω_y
(4,5)	ω_{in_z}	(4,6)	$-\omega_{in_y}$
(4,8)	$-C_{RY}$	(5,1)	Ω_z
(5,3)	$-\Omega_x$	(5,4)	$-\omega_{in_z}$
(5,6)	ω_{in_x}	(5,7)	C_{RX}
(6,1)	$-\Omega_y$	(6,2)	Ω_x
(6,4)	ω_{in_y}	(6,5)	$-\omega_{in_x}$
(7,1)	$-2V_y\Omega_y - 2V_z\Omega_z$	(7,2)	$2V_y\Omega_x$
(7,3)	$2V_z\Omega_y$	(7,5)	$-A_z$
(7,6)	A_y	(7,7)	$-V_zC_{RX}$
(7,8)	$2\Omega_z$	(7,9)	$-\rho_y - 2\Omega_y$
(8,1)	$2V_x\Omega_y$	(8,2)	$-2V_x\Omega_x - 2V_z\Omega_z$
(8,3)	$2v_z\Omega_y$	(8,4)	A_z
(8,6)	$-A_x$	(8,7)	$-2\Omega_z$
(8,8)	$-V_zC_{RY}$	(8,9)	$\rho_x + 2\Omega_x$
(9,1)	$2V_x\Omega_z$	(9,2)	$2V_y\Omega_z$
(9,3)	$-2V_y\Omega_y - 2V_x\Omega_x$	(9,4)	$-A_y$
(9,5)	A_x	(9,7)	$\rho_y + 2\Omega_y + V_xC_{RX}$
(9,8)	$-\rho_x - 2\Omega_x + V_yC_{RY}$	(9,10)	$2g_o/a$
(10,9)	1		

$\rho_{x,y}$ = Components of angular rate, nav reference frame to earth-fixed frame
$\Omega_{x,y,z}$ = Components of angular rate, earth-fixed frame to inertial frame
$\omega_{in_{x,y,z}}$ = Components of angular rate, nav reference frame to inertial frame
$V_{x,y,z}$ = Components of vehicle velocity vector in earth-fixed coordinates
$A_{x,y,z}$ = Components of specific force in the sensor reference frame
$C_{RX,RY}$ = Components of earth spheroid inverse radii of curvature
g_0 = Equatorial gravity magnitude (32.08744 $ft/\sec^2$)
a = Equatorial radius of the earth (6378388 m)

Table 24. Elements of the Dynamics Submatrix $\mathbf{F}_{(red)12}$

Element	Term	Element	Term
(9,11)	k_2	(10,11)	k_1

Table 25. Elements of the Dynamics Submatrix $\mathbf{F}_{(red)22}$

Element	Term
(11,11)	$-\beta_{\delta h_c}$

Table 26. Elements of Process Noise Submatrix $\mathbf{Q}_{(red)11}$

Element	Term	Element	Term
(4,4)	$Q_{\eta_{b_x}}$	(7,7)	$Q_{\eta_{A_x}}$
(5,5)	$Q_{\eta_{b_y}}$	(8,8)	$Q_{\eta_{A_y}}$
(6,6)	$Q_{\eta_{b_z}}$	(9,9)	$Q_{\eta_{A_z}}$

Table 27. Elements of Process Noise Submatrix $\mathbf{Q}_{(red)22}$

Element	Term
(11,11)	$2\beta_{\delta h_c}\sigma^2_{\delta h_c}$

$k_{1,2}$ = Vertical channel gains, see LN-93 documentation [25] for equations
$\beta_{\delta h_c}$ = Barometer inverse correlation time ($\tau = 10$ min; $\beta = \frac{1}{\tau}$)
$Q_{\eta_{b_{x,y,z}}}$ = PSD value of gyro drift rate white noise (6.25e-10$\frac{deg^2}{sec^3}$)
$Q_{\eta_{A_{x,y,z}}}$ = PSD value of accelerometer white noise (1.037e-7$\frac{ft^2}{sec^3}$)
$\sigma^2_{\delta h_c}$ = Variance of barometric altimeter correlated noise ($10000 ft^2$)

Bibliography

[1] Abramson, P. D., Jr. *Simultaneous Estimation of the State and Noise Statistics in Linear Dynamic Systems*. PhD dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts, Cambridge, Massachusetts, May 1968.

[2] Athans, M., et al. "The Stochastic Control of the F-8C Aircraft Using a Multiple Model Adaptive Control (MMAC) Method - Part I: Equilibrium Flight," *IEEE Transactions on Automatic Control*, Vol. AC-22, No. 5, 768–780, October 1977.

[3] Athans, M. and C. B. Chang. *Adaptive Estimation and Parameter Identification Using a Multiple Model Estimation Algorithm*. Technical Note 1976-28, Lexington, MA: Lincoln Lab., MIT, June 1976. ESD-TR-76-184.

[4] Baram, Yoram. *Information, Consistent Estimation and Dynamic System Identification*. PhD dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts, Cambridge, Massachusetts, November 1976.

[5] Blom, Henk A. P. and Yaakov Bar-Shalom. "The Interacting Multiple Model Algorithm for Systems with Markovian Switching Coefficients," *IEEE Transactions on Automatic Control*, Vol. 33, No. 8, 780–783, August 1988.

[6] Britting, Kenneth R. *Inertial Navigation Systems Analysis*. New York: Wiley-Interscience, 1971.

[7] Britton, Ryan, L. *A Differential GPS-aided INS for Aircraft Landings*. MS thesis, AFIT/GE/ENG/95D-03, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1995.

[8] Carlson, Neal A. "Federated Filter for Fault-Tolerant Integrated Navigation," *Aerospace Navigation Systems*, Advisory Group for Aerospace Research and Development, AGARDOgraph Article 331, Neuilly-Sur-Seine France, June 1995.

[9] Carlson, Neal A. *Distributed Kalman Filter Architectures Phase II*. Draft Final Report, WL/AAAI, Wright-Patterson AFB, OH, April 1995.

[10] Carlson, Neal A. and Jr. C. M. Neily. *Distributed Kalman Filter Architectures*. Final Technical Report, AFWAL-TR-87-1181, Avionics Laboratory, Wright-Patterson AFB, OH, June 1987.

[11] Chang, C. B. and M. Athans. "State Estimation for Discrete Systems with Switching Parameters," *IEEE Transactions on Aerospace and Electronic Systems*, AES–14, No. 4, 418–424, May 1978.

[12] Clark, Curtis S. *Multiple Model Adaptive Estimation and Control Redistribution for the Vista F-16 During Partial Actuator Impairments*. MS thesis, AFIT/GE/ENG/97D-23, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1997.

[13] Deckert, James C., et al. "F-8 DFBW Sensor Failure Identification Using Analytic Redundancy," *IEEE Transactions on Automatic Control*, Vol. AC-22, No. 5, 795–803, October 1977.

[14] Eide, Peter K. *Implementation and Demonstration of a Multiple Model Adaptive Estimator*

Failure Detection System for the F-16. MS thesis, AFIT/GE/ENG/94D-06, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1994.

[15] Eide, Peter K. and Peter S. Maybeck. "An MMAE Failure Detection System for the F-16," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 32, No. 3, 1125–1148, July 1996.

[16] Gray, Robert A. *An Integrated GPS/INS/BARO and Radar Altimeter System for Aircraft Precision Approach Landings*. MS thesis, AFIT/GE/ENG/94D-13, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1994.

[17] Griffin, Gordon C. *Control of a Large Space Structure Using Multiple Model Adaptive Estimation and Control Techniques*. MS thesis, AFIT/GE/ENG/94D-14, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1994.

[18] Griffin, Gordon C. and Peter S. Maybeck. "MMAE/MMAC Techniques Applied to Large Space Structure Bending with Multiple Uncertain Parameters," *Proceeding of the 34th IEEE Conference on Decision and Control*, New Orleans, LA, 1153–1158, December 1995.

[19] Gustafson, John A. *Control of a Large Flexible Space Structure Using Multiple Model Adaptive Algorithms*. MS thesis, AFIT/GE/ENG/91D-22, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1991.

[20] Gustafson, John A. and Peter S. Maybeck. "Flexible Spacestructure Control Via Moving-Bank Multiple Model Algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 30, No. 3, 750–757, July 1994.

[21] Hanlon, Peter D. *Failure Identification Using Multiple Model Adaptive Estimaton for the LAMBDA Flight Vehicle*. MS thesis, AFIT/GE/ENG/92D-19, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1992.

[22] Hanlon, Peter D. *Practical Implementation of Multiple Model Adaptive Estimation Using Neyman-Pearson Based Hypothesis Testing and Spectral Estimation Tools*. PhD dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, Wright-Patterson AFB, OH, September 1996.

[23] Hentz, K. P. *Feasibility Analysis of Moving Bank Multiple Model Adaptive Estimation and Control Algorithms*. MS thesis, AFIT/EE/ENG/84D-32, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1984.

[24] Honeywell Military Avionics Division, Minneapolis, MN. *Honeywell AN/APN-194 Pulse Radar Altimeter System*, June 1989. Honeywell Technical Description.

[25] Knudsen, L. *Performance Accuracy (Truth Model/Error Budget) Analysis for the LN-93 Inertial Navigation System Inertial Navigation Unit*. Technical Report, 5500 Canoga Avenue, Woodland Hills, California 91365: Litton Guidance and Control Systems, January 1985. DID No. Di-S-21433 B/T: CDRL No. 1002.

[26] Kyger, David W. *Reducing Lag in Virtual Displays Using Multiple Model Adaptive Estimation*. MS thesis, AFIT/GE/ENG/95D-11, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1995.

[27] Kyger, David W. and Peter S. Maybeck. "Reducing Lag in Virtual Displays Using Multiple Model Adaptive Estimation." Accepted for publication in *IEEE Transactions on AES*, October 1998.

[28] Lewantowicz, Z. H. and D. W. Keen. "Graceful Degradation of GPS/INS Performance with Fewer Than Four Satellites," *The Institute of Navigation, National Technical Meeting*, Phoenix, AZ, 269–275, January 1991.

[29] Lewis, Robert W. *Multiple Model Adaptive Estimation and Control Redistribution for the VISTA F-16*. MS thesis, AFIT/GE/ENG/96D-29, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1996.

[30] Li, Xiao-Rong, et. al. "Multiple-Model Estimation with Variable Structure: Model-Group Switching Algorithm," *Proceedings of the 36th IEEE Conference on Decision and Control*, San Diego, CA, 3114–3119, December 1997.

[31] Li, Xiao-Rong and Yaakov Bar-Shalom. "Multiple-Model Estimation with Variable Structure," *IEEE Transactions on Automatic Control*, Vol. 41, No. 4, 479–493, April 1996.

[32] Libby, E. W. *Application of Sequence Comparison Methods to Multisensor Data Fusion and Target Recognition*. PhD dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, June 1993.

[33] Libby, E. W. and Peter S. Maybeck. "Sequence Comparison Techniques for Multisensor Data Fusion and Target Recognition," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 32, No. 1, 52–65, January 1996.

[34] Lund, Elvind J. *On-line Discrimination and Estimation in Multiple Regime Systems*. PhD dissertation, Division of Engineering Cybernetics, The Norwegian Institute of Technology University of Trondheim, Trondheim, Norway, June 1992.

[35] Lund, Elvind J., et al. "Multiple Model Estimation with Inter-Residual Distance Feedback," *Modeling, Identification and Control*, Vol. 13, No. 3, 127–140, 1992.

[36] Magill, D. T. "Optimal Adaptive Estimation of Sampled Stochastic Processes," *IEEE Transactions on Automatic Control*, AC–10, No. 5, 434–439, 1965.

[37] Martin, Daniel P. and Neal A. Carlson. *Distributed Model Adaptive Estimation Phase I Final Report*. Final Report, TR-91-003, WL/AAAN-2, Wright-Patterson AFB, OH, August 1991.

[38] Martin, E. H. "GPS User Equipment Error Models," *The Institute of Navigation*, 1, 109–118 1980.

[39] The MathWorks, Inc., Natick, MA. *Optimization Toolbox for Use with MATLAB (registered trademark)*, December 1992.

[40] The MathWorks, Inc., Natick, MA. *MATLAB (registered trademark)* (Version 4.2c Edition), November 1994.

[41] Maybeck, Peter S. "Adaptive Tracking of Maneuvering Targets Based on IR Image Data," *AGARD Lecture Series No. 166*, published in London, England, 7, 1–18 July 1989.

[42] Maybeck, Peter S. *Combined State and Parameter Estimation for On-line Applications*. PhD dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts,

Cambridge, Massachusetts, February 1972.

[43] Maybeck, Peter S. *Stochastic Models, Estimation, and Control*, I. New York: Academic Press, Inc., 1979. Republished, Arlington, VA: Navtech, 1994.

[44] Maybeck, Peter S. *Stochastic Models, Estimation, and Control*, II. New York: Academic Press, Inc., 1982. Republished, Arlington, VA: Navtech, 1994.

[45] Maybeck, Peter S. *Stochastic Models, Estimation, and Control*, III. New York: Academic Press, Inc., 1982.

[46] Maybeck, Peter S. "Moving-Bank Multiple Model Adaptive Estimation and Control Algorithms: An Evaluation," *Control and Dynamics Systems*, Vol. 31, 1989. Edited by C. T. Leondes, Academic Press, NY.

[47] Maybeck, Peter S. and Peter D. Hanlon. "Performance Enhancement of a Multiple Model Adaptive Estimator," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 31, No. 4, 1240–1254, October 1995.

[48] Maybeck, Peter S. and K. P. Hentz. "Investigation of Moving-Bank Multiple Model Adaptive Algorithms," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 10, No. 1, 90–96, 1987.

[49] Maybeck, Peter S. and D.L. Pogoda. "Multiple Model Adaptive Controller for STOL F-15 with Sensor/Actuator Failures," *Proceedings of the 28th IEEE Conference on Decision and Control*, 1566–1572, Tampa, Florida, December 1989.

[50] Maybeck, Peter S. and Richard D. Stevens. "Reconfigurable Flight Control Via Multiple Model Adaptive Control Methods," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 27, No. 3, 470–479, May 1991.

[51] Meer, D.E. and P. S. Maybeck. "Multiple Model Adaptive Estimation for Space-Time Point Process Observations," *Proceedings of the 23rd IEEE Conference on Decision and Control*, Las Vegas, NV, December 1984.

[52] Mendenhall, William., et al. *Mathematical Statistics with Applications, Second Edition*. University of Florida: Duxbury Press, 1981.

[53] Menke, Timothy E. *Multiple Model Adaptive Estimation Applied to the VISTA F-16 with Actuator and Sensor Failures*. MS thesis, AFIT/GE/ENG/92J-01, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, June 1992.

[54] Menke, Timothy E. and Peter S. Maybeck. "Sensor/Actuator Failure Detection in the VISTA F-16 by Multiple Model Adaptive Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 31, No. 4, 1218–1228, October 1995.

[55] Moose, Richard L. and P. P. Wang. "An Adaptive Estimator with Learning for a Plant Containing Semi-Markov Switching Parameters," *IEEE Transactions on Systems, Man., and Cyber*, Vol. 3, No. 3, 277–281, May 1973.

[56] Mosle, William B. *Detection, Isolation, and Recovery of Failures in an Integrated Navigation System*. MS thesis, AFIT/GE/ENG/93D-28, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1993.

[57] Muravez, Randall J. *Multiple Model Adaptive Estimation and Prediction with the*

Harmonically Balanced Kalman Filter Bank. MS thesis, California Polytechnic University, Pomona, CA, December 1989.

[58] Musick, Stanton H. *PROFGEN – A Computer Program for Generating Flight Profiles*. Technical Report, Wright-Patterson AFB, OH: Air Force Avionics Laboratory, November 1976. AFAL-TR-76-247.

[59] Musick, Stanton H. *MSOFE – Multimode Simulation for Optimal Filter Evaluation*. Technical Report, Wright-Patterson AFB, OH: Air Force Avionics Laboratory, October 1980. AFWAL-TR-88-1138.

[60] Musick, Stanton H. and Neil Carlson. *User's Manual for a Multimode Simulation for Optimal Filter Evaluation (MSOFE)*. Air Force Avionics Laboratory, Wright-Patterson AFB, OH, April 1990. AFWAL-TR-88-1136.

[61] Negast, William J. *Incorporation of Differential Global Positioning System Measurements Using an Extended Kalman Filter for Improved Reference System Performance*. MS thesis, AFIT/GE/ENG/91D-41, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1991.

[62] Nielsen, Robert L. *Development of a Performance Evaluation Tool (MMSOFE) for Detection of Failures with Multiple Model Adaptive Estimation (MMAE)*. MS thesis, AFIT/GE/ENG/93S-37, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1993.

[63] Nielsen, Robert L. and Stanton H. Musick. *MMSOFE – Multiple Model Simulation for Optimal Filter Evaluation*. Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1993.

[64] Pervan, Boris S., et. al. "Integrity Monitoring for Precision Approach Using Kinematic GPS and a Ground-Based Pseudolite," *Navigation Journal of the ION*, Vol. 41, No.2, 159–173 January, 1994.

[65] Pinson, J. C. "Inertial Guidance for Cruise Vehicles," *in Guidance and Control of Aerospace Vehicles* (C. T. Leondes, ed.). McGraw-Hill, NY, 1963.

[66] Riggins, Robert N. Jr. *Detection and Isolation of Plant Failures in Dynamic Systems*. PhD dissertation, University of Michigan, 1991.

[67] Schiller, Gregory J. *Control of a Large Space Structure Using Multiple Model Adaptive Estimation and Control Techniques*. MS thesis, AFIT/GE/ENG/93D-02, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1993.

[68] Schiller, Gregory J. and Peter S. Maybeck. "Space Structure Control Using Multiple Model Adaptive Estimation and Control," *Proceeding of the 13th IFAC Symposium Automatic Control in Aerospace-Aerospace Control '94,* September 1994.

[69] Sheldon, Stuart N. *An Optimal Parameter Discretization Strategy for Multiple Model Adaptive Estimation and Control*. PhD dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1989.

[70] Sheldon, Stuart N. and Peter S. Maybeck. "An Optimizing Design Strategy for Multiple Model Adaptive Estimation and Control," *IEEE Transactions on Automatic Control*, Vol.

38, No. 4, 651–654, April 1993.

[71] Stepaniak, Michael J. *Multiple Model Adaptive Control of the VISTA F-16*. MS thesis, AFIT/GE/ENG/95D-26, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1995.

[72] Stepaniak, Michael J. and Peter S. Maybeck. "MMAE-Based Control Redistribution Applied to the VISTA F-16." Accepted for publication in *IEEE Transactions on AES*, October 1998.

[73] Stevens, Richard D. *Characterization of a Reconfigurable Multiple Model Adaptive Controller Using a STOL F-15 Model*. MS thesis, AFIT/GE/ENG/89D-52, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1989.

[74] Van Trees, H. L. *Detection, Estimation and Modulation Theory*. New York: Wiley and Sons, 1968.

[75] Vasquez, Juan R. "Moving-Bank Multiple Model Adaptive Estimation and Failure Detection." PhD Research Prospectus, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, Unpublished.

[76] Vasquez, Juan R. *Detection of Spoofing, Jamming, or Failure of a Global Positioning System (GPS)*. MS thesis, AFIT/GE/ENG/92D-37, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1992.

[77] Vasquez, Juan R. *New Algorithms for Moving-Bank Multiple Model Adaptive Estimation*. PhD dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, Wright-Patterson AFB, OH, In Progress, 1998.

[78] White, Nathan A. *MMAE Detection of Interference/Jamming and Spoofing in a DGPS-Aided INS*. MS thesis, AFIT/GE/ENG/96D-21, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1996.

[79] White, N. A., P. S. Maybeck and S.L. DeVilbiss. "Detection of Interference/Jamming and Spoofing in a DGPS-Aided Inertial System." Accepted for publication in *IEEE Transactions on AES*, October 1998.

[80] Willsky, A. S. "A Survey of Design Methods for Failure Detection in Dynamic Systems," *Automatica*, No. 12, 601–611, December 1976.

[81] Willsky, A. S. and H. Jones. *A Generalized Likelihood Ratio Approach to State Estimation in Linear Systems Subject to Abrupt Changes*. Technical Report, Reading, MA: The Analytic Sciences Corporation, 1976. Under USAF Contract No. F04701-74-C-0095.

[82] Zicker, William L. *Pointing and Tracking of Particle Beams*. MS thesis, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1983. AD-A138 311.

Vita

Major Mikel M. Miller was born in Sioux City Iowa on May 8, 1960. In 1978 he entered North Dakota State University where he received the degree of Bachelor of Science in Electrical and Electronic Engineering, graduating in November 1992. He received his commission through the Air Force ROTC program in which he was a Distinguished Graduate. His first assignment was at Offutt AFB in Omaha, Nebraska, where he served as a Satellite Systems Engineer in the 1000th Satellite Operations Group. His second assignment was at Wright-Patterson AFB (WPAFB) in Dayton, Ohio, where he earned his Masters of Science degree in Electrical Engineering from the Air Force Institute of Technology (AFIT). His third assignment was at Holloman AFB in Alamogordo, New Mexico, where he served as a Program Manager for the Central Inertial Guidance and Test Facility. His fourth assignment was at Honeywell Inc. in Clearwater, Florida, where he participated in the Air Force's Education With Industry program. His fifth assignment was back at WPAFB, where he served as Chief, Plans and Integration Section for the F-22 System Program Office. In June 1994, he entered the School of Engineering, AFIT, for a PhD program in Electrical Engineering. He is member of Tau Beta Pi and Eta Kappa Nu and upon graduation he will join the faculty at AFIT and serve as an Assistant Professor in Electrical Engineering. He is blessed to be married to the former Colleen Marie Conlin of Williston, ND, and to have four fantastic children – Casey, Krista, Trevor, and Megan.

Permanent Address:
4208 Spruce Wood Drive
Beavercreek, OH 45432

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1998	Doctoral Dissertation

4. TITLE AND SUBTITLE
MODIFIED MULTIPLE MODEL ADAPTIVE ESTIMATION (M3AE) FOR SIMULTANEOUS PARAMETER AND STATE ESTIMATION

5. FUNDING NUMBERS

6. AUTHOR(S)
Mikel M. Miller, Major, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/DS/ENG/98-02

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Ms. Sandra Berning
AFRL/SNAR
2241 Avionics Circle, WPAFB OH 45433-7318

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In many estimation problems, it is desired to estimate system states and parameters simultaneously. However, inherent to traditional estimation architectures of the past, the designer has had to make a trade-off decision between designs intended for accurate state estimation versus designs concerned with accurate parameter estimation. This research develops one solution to this trade-off decision by proposing a new architecture based on Kalman filtering (KF) and Multiple Model Adaptive Estimation (MMAE) techniques. This new architecture, the Modified-MMAE (M3AE), exploits the benefits of an MMAE designed for accurate parameter estimation, and yet performs at least as well in state estimation as an MMAE designed for accurate state estimation. The M3AE accomplishes the simultaneous estimation task by providing accurate state estimates from a single KF designed to accept accurate parameter estimates from the MMAE. Additionally, an M3AE approximate covariance analysis capability is developed, giving the designer a valuable design tool for analyzing and predicting M3AE performance before actually implementing the M3AE and conducting a time-consuming full-scale Monte Carlo performance analysis. Finally, the M3AE architecture is applied to two existing research examples to demonstrate the performance improvement over that of conventional MMAEs. The first example involves a simple second-order mechanical translational system, in which the system's natural frequency is the uncertain parameter. The second example involves a 13-state nonlinear integrated Global Positioning System/Inertial Navigation System (GPS/INS) system, in which the variance of the measurement noise affecting the GPS outputs, is the uncertain parameter.

14. SUBJECT TERMS		15. NUMBER OF PAGES
Kalman Filtering	State Estimation	241
Multiple Model Adaptive Estimation	Covariance Analysis	16. PRICE CODE
Parameter Estimation		

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL